



Best Practice & Design Pattern  
pour les langages du Web  
sémantique

[olivier.corby@inria.fr](mailto:olivier.corby@inria.fr)

Université Côte d'Azur  
Inria, I3S  
Wimmics

# Plan

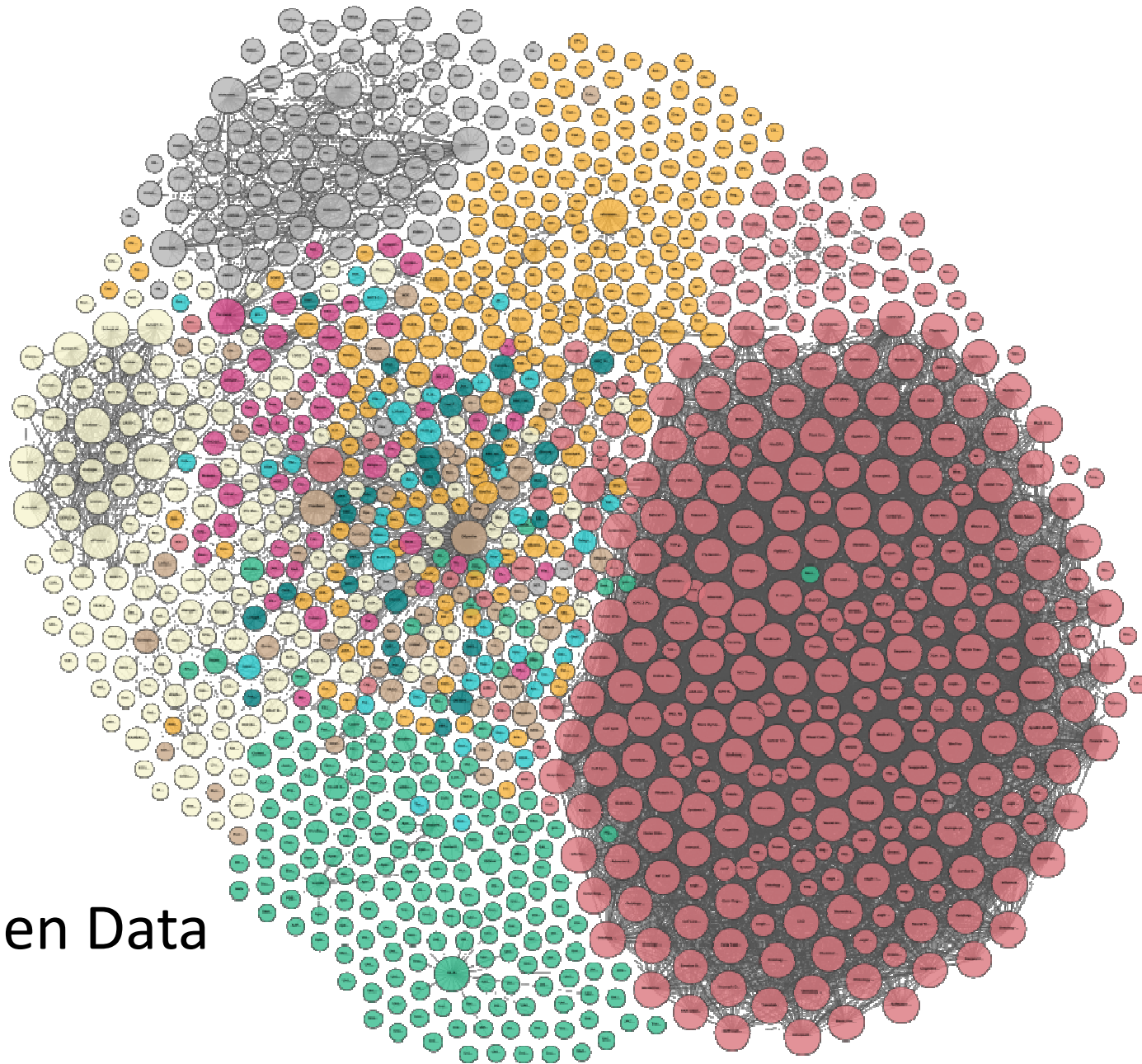
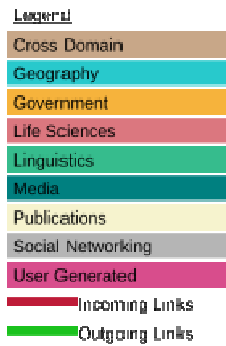
1. RDF : Resource Description Framework
2. RDFS : RDF Schema
3. SPARQL : RDF Query Language

- <http://wimmics.inria.fr/lectures>

# Motto

RDF c'est simple ...

... et cela permet de faire des choses complexes



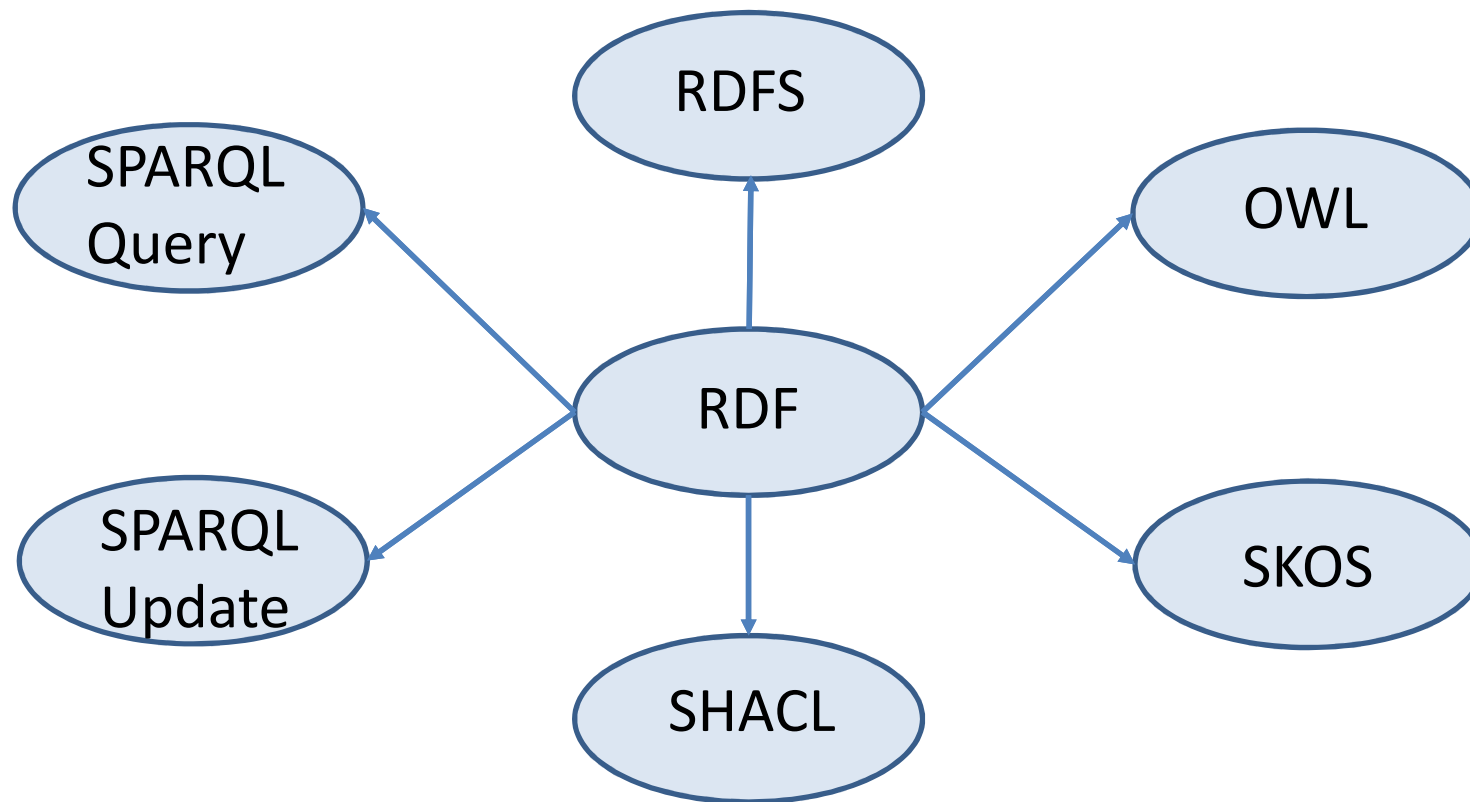
# Linked Open Data Cloud

SPARQL Template Transformation Language

4

"Linking Open Data cloud diagram 2017, by Andrejs Abele, John P. McCrae, Paul Buitelaar, Anja Jentzsch and Richard Cyganiak. <http://lod-cloud.net/>"

# Ecosystème RDF



# RDF

1. Relation n-aire
2. Liste
3. Dataset
4. AST

# RDF

- Graphe orienté étiqueté
- Triple
  - subject property object
- RDF term
  - URI
  - Blank Node
  - Literal
    - XML Schema Datatype

# RDF Syntax

- **Turtle**
- RDF/XML
- RDFa
- Trig
- JSON-LD



# Triple : relation binaire

subject property object

us:mobile us:move 100

move(mobile, 100)

# Relation n-aire

- La vitesse du mobile est 100 km/h

`move(mobile, 100, km/h)`

- John est à Marseille le 5/7/2017

`location(John, Marseille, 2017-7-5)`

# « Triple » n-aire ?

- La vitesse du mobile est 100 km/h

mobile move 100 -- km/h

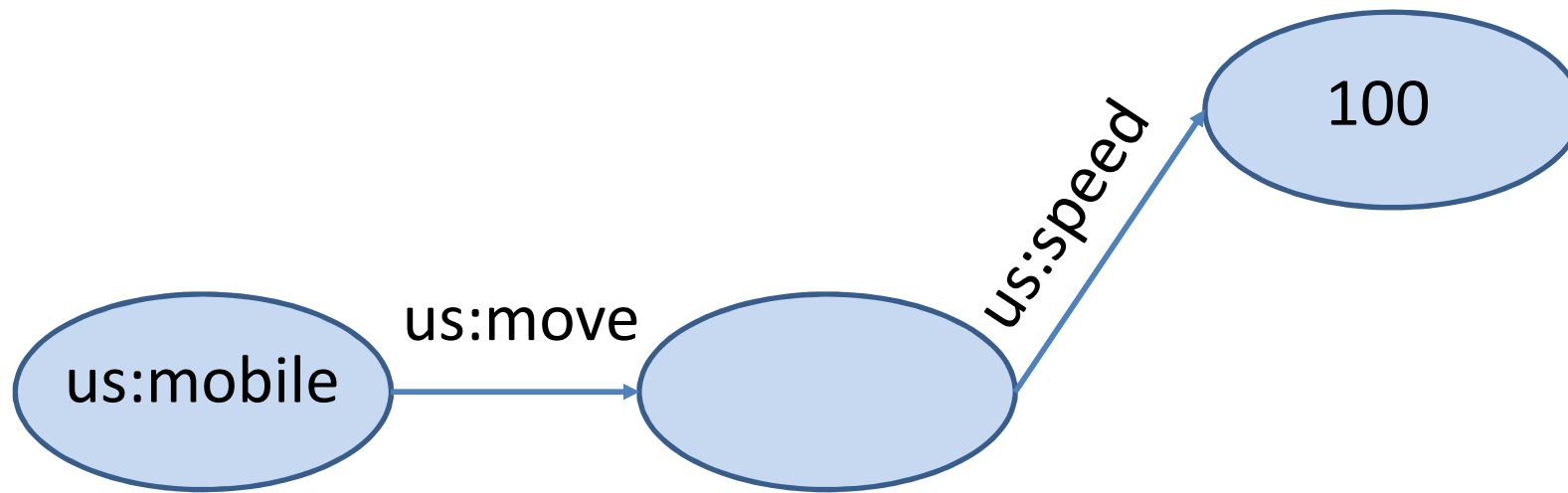
- John est à Marseille le 5/7/2017

John location Marseille -- 2017-7-5

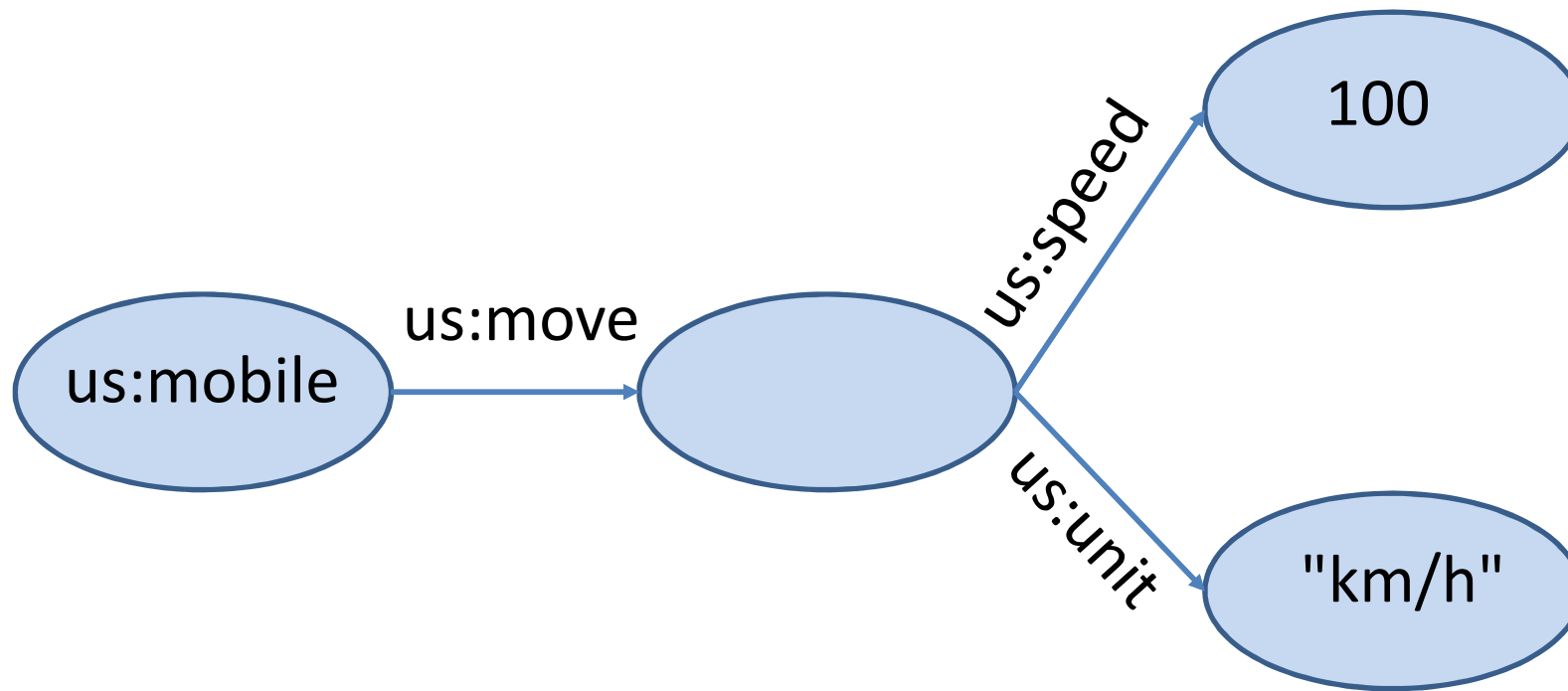
# Relation n-aire (1)



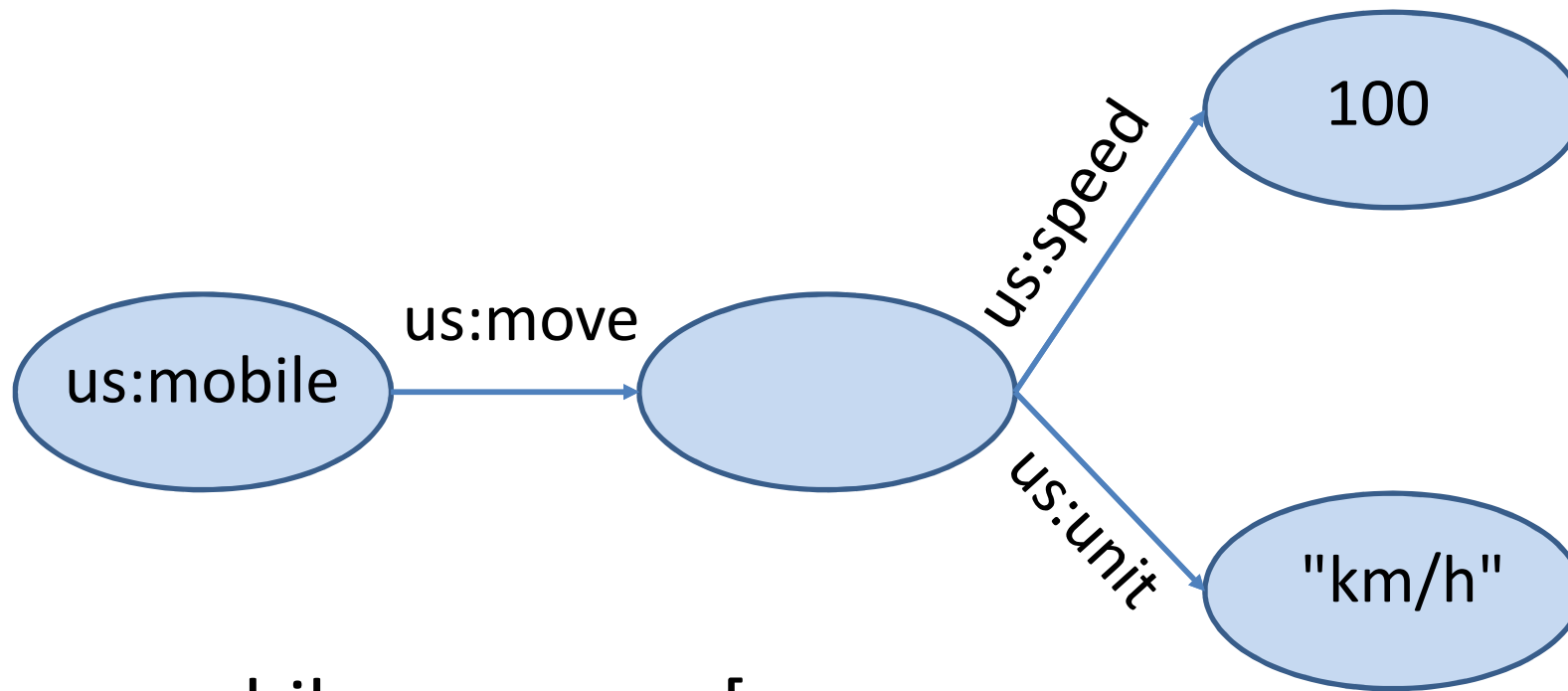
# Relation n-aire (1)



# Relation n-aire (1)

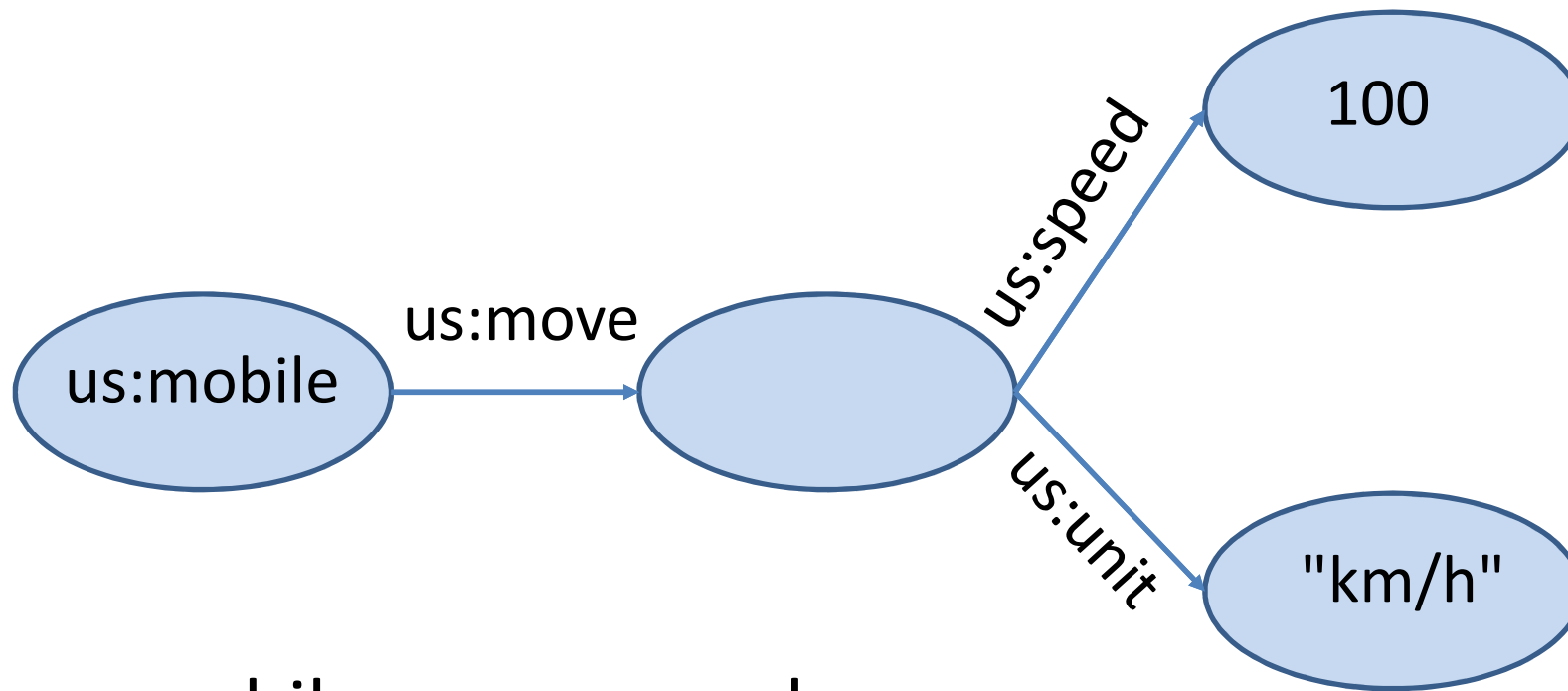


# Relation n-aire (1)



us:mobile us:move [  
us:speed 100 ;  
us:unit "km/h" ] .

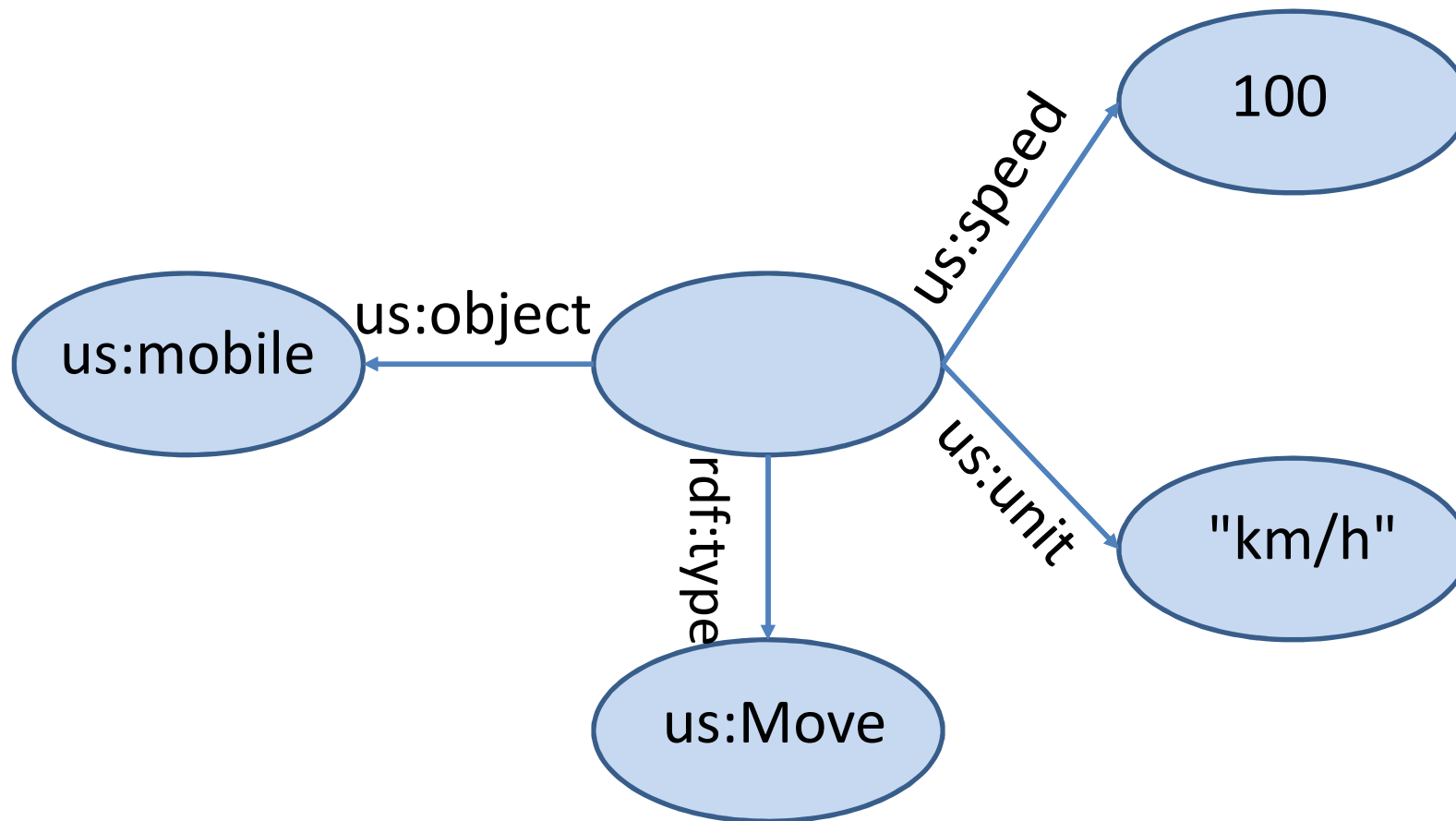
# Relation n-aire (1)



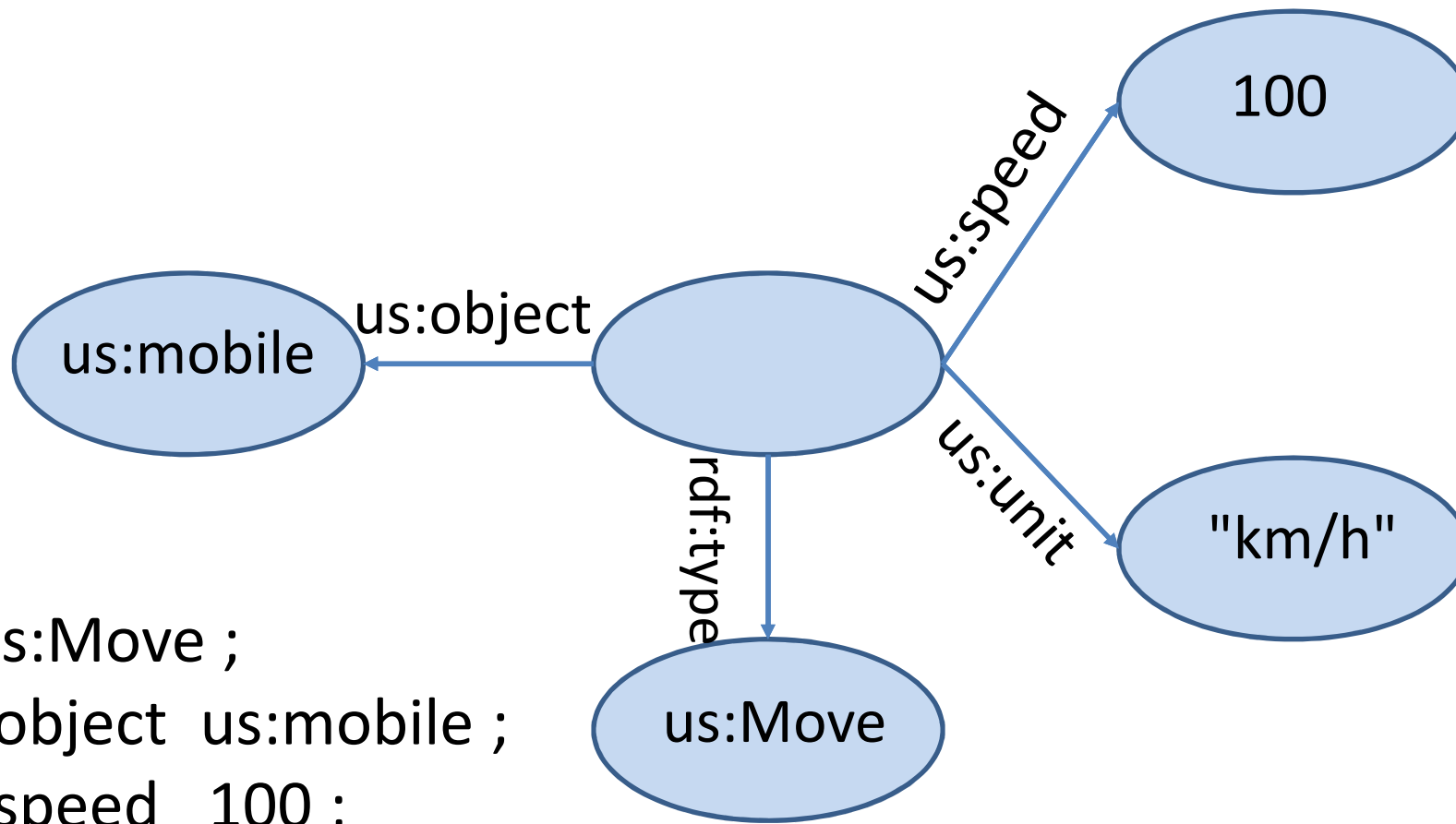
us:mobile us:move \_:b .  
\_:b us:speed 100 ;  
us:unit "km/h" .



## Relation n-aire (2)



## Relation n-aire (2)

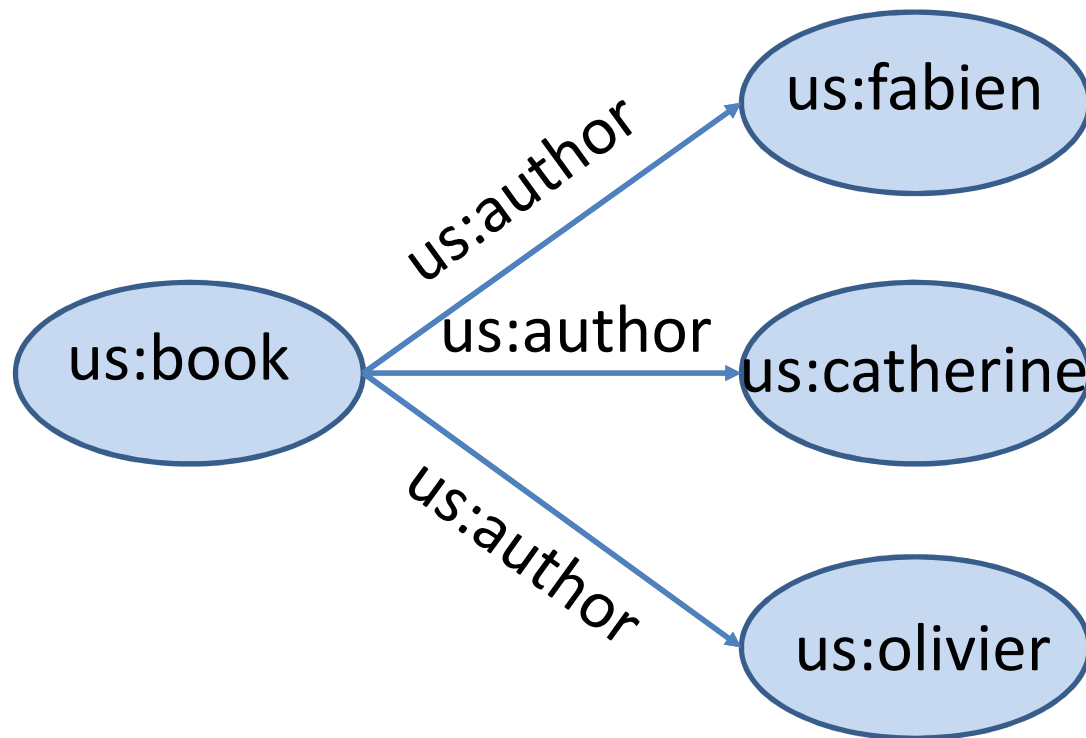


```
[ a us:Move ;  
  us:object us:mobile ;  
  us:speed 100 ;  
  us:unit "km/h" ] .
```

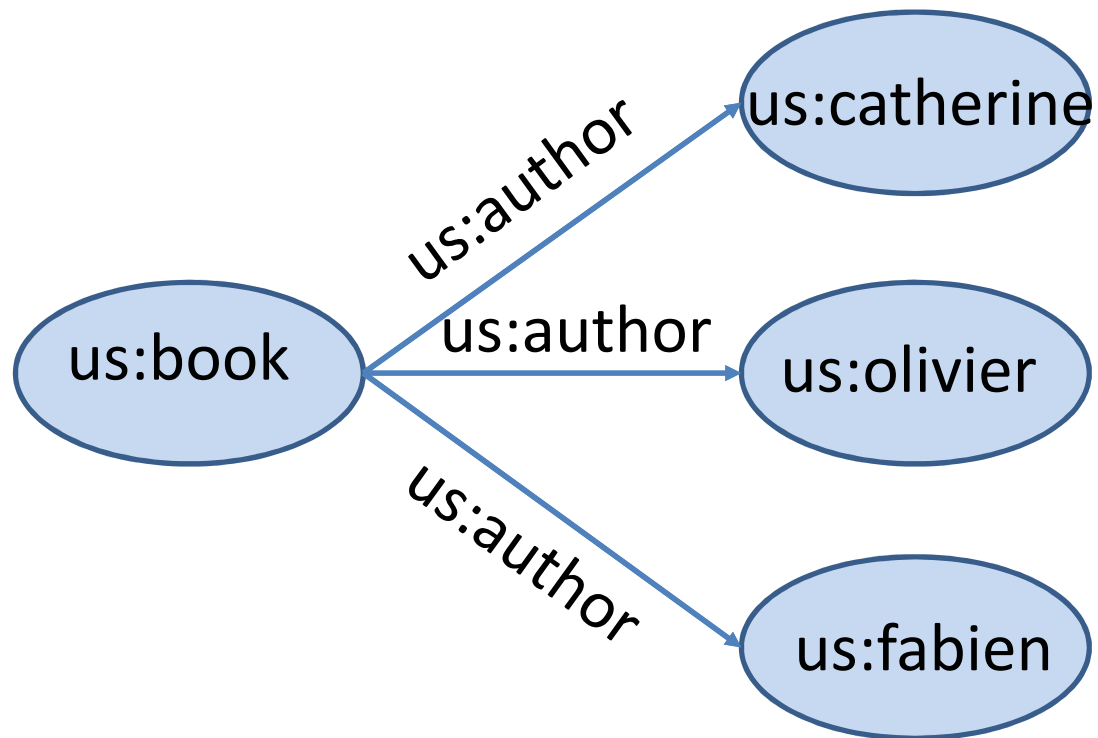
## Relation n-aire (3)

us:mobile us:speed "100"^^unit:kmph

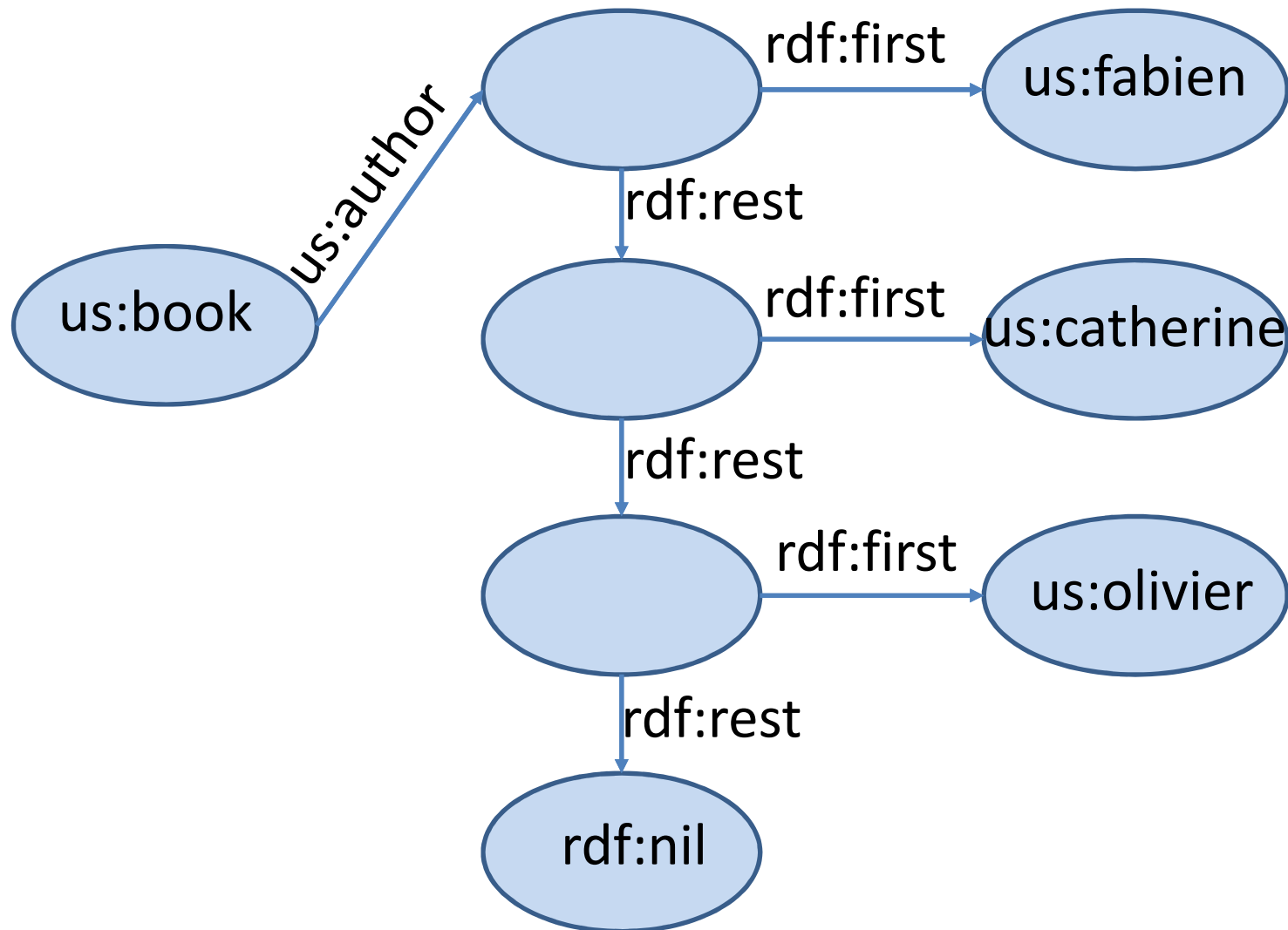
# Pas d'ordre sur les arcs



# Pas d'ordre sur les arcs



# Structure de liste



# Liste

us:book us:author (us:fabien us:catherine us:olivier)

# Liste

us:book us:author (us:fabien us:catherine us:olivier)

us:book us:author \_:b1 .

\_:b1 rdf:first us:fabien .

\_:b1 rdf:rest \_:b2 .

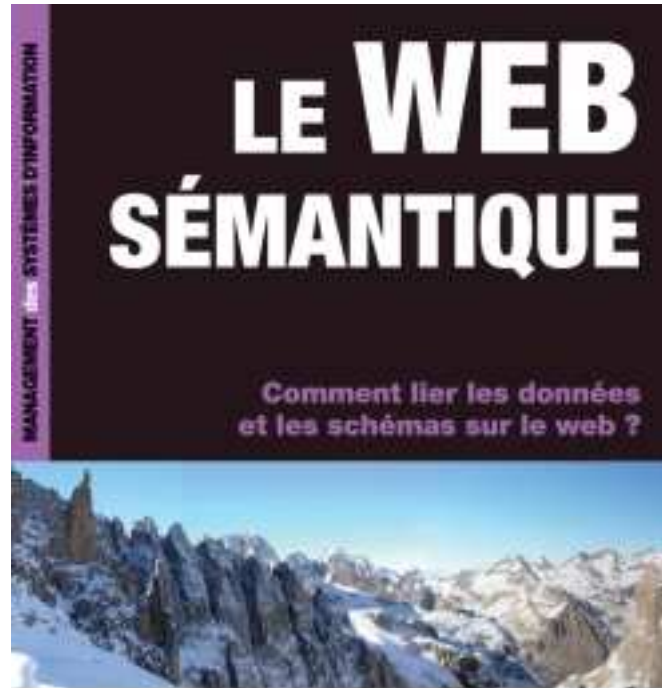
\_:b2 rdf:first us:catherine .

\_:b2 rdf:rest \_:b3 .

\_:b3 rdf:first us:olivier .

\_:b3 rdf:rest rdf:nil .





Fabien Gandon  
Catherine Faron-Zucker  
Olivier Corby

DUNOD

# RDF Dataset

- Graphe par défaut (default graph)
- Graphes nommés (named graph)

$\{ G, (uri_1 \rightarrow G_1), .. (uri_n \rightarrow G_n) \}$

# Graphe par défaut

```
ex:James a ex:Lecturer ;  
    foaf:name "James" .
```

```
ex:James a ex:Musician ;  
    foaf:name "Jimmy" .
```

# Graphes nommés

```
graph ex:g1 {  
    ex:James a ex:Lecturer ;  
    foaf:name "James" .  
}
```

```
graph ex:g2 {  
    ex:James a ex:Musician ;  
    foaf:name "Jimmy" .  
}
```

# Graphes nommés

```
graph ex:g1 {  
    ex:James a ex:Lecturer ;  
    foaf:name "James" .  
}
```

```
graph ex:g2 {  
    ex:James a ex:Musician ;  
    foaf:name "Jimmy" .  
}
```

# Annoter URI de graphe nommé

```
graph ex:g1 {  
    ex:James a ex:Lecturer ;  
    foaf:name "James" .  
}
```

# Annoter URI de graphe nommé

```
graph ex:g1 {  
    ex:James a ex:Lecturer ;  
    foaf:name "James" .  
}
```

ex:g1 property value .

# Annoter URI de graphe nommé

```
graph ex:g1 {  
    ex:James a ex:Lecturer ;  
    foaf:name "James" .  
}
```

```
ex:g1 ex:date "1930-01-29"^^xsd:date ;  
    ex:author ex:John .
```



# Typing URI de graphe nommé

```
graph ex:g1 {  
    ex:James a ex:Lecturer ;  
    foaf:name "James" .  
}
```

```
ex:g1 a ex:Context .
```

# Annoter URI de graphe nommé

```
graph ex:g1 { ... }
```

```
graph ex:g2 { ... }
```

```
ex:g1 ex:before ex:g2 .
```

# Graphes nommés

- Contextualiser les données
- Annoter les URI des graphes nommés
- Modéliser la provenance des données
- Version
- Annotation temporelle

# Graphes nommés

- *Les graphes nommés n'ont pas de sémantique !*
- *La relation entre URI et graphe nommé n'a pas de sémantique !*
- *C'est pas très grave ... mais ça limite l'interopérabilité*

# Abstract Syntax Tree

# Abstract Syntax Tree

- RDF peut représenter des arbres de syntaxe
- Pour annoter des données avec des expressions
- SPIN : format RDF pour SPARQL
- OWL/RDF : format RDF pour OWL
- SHACL : format RDF pour schema

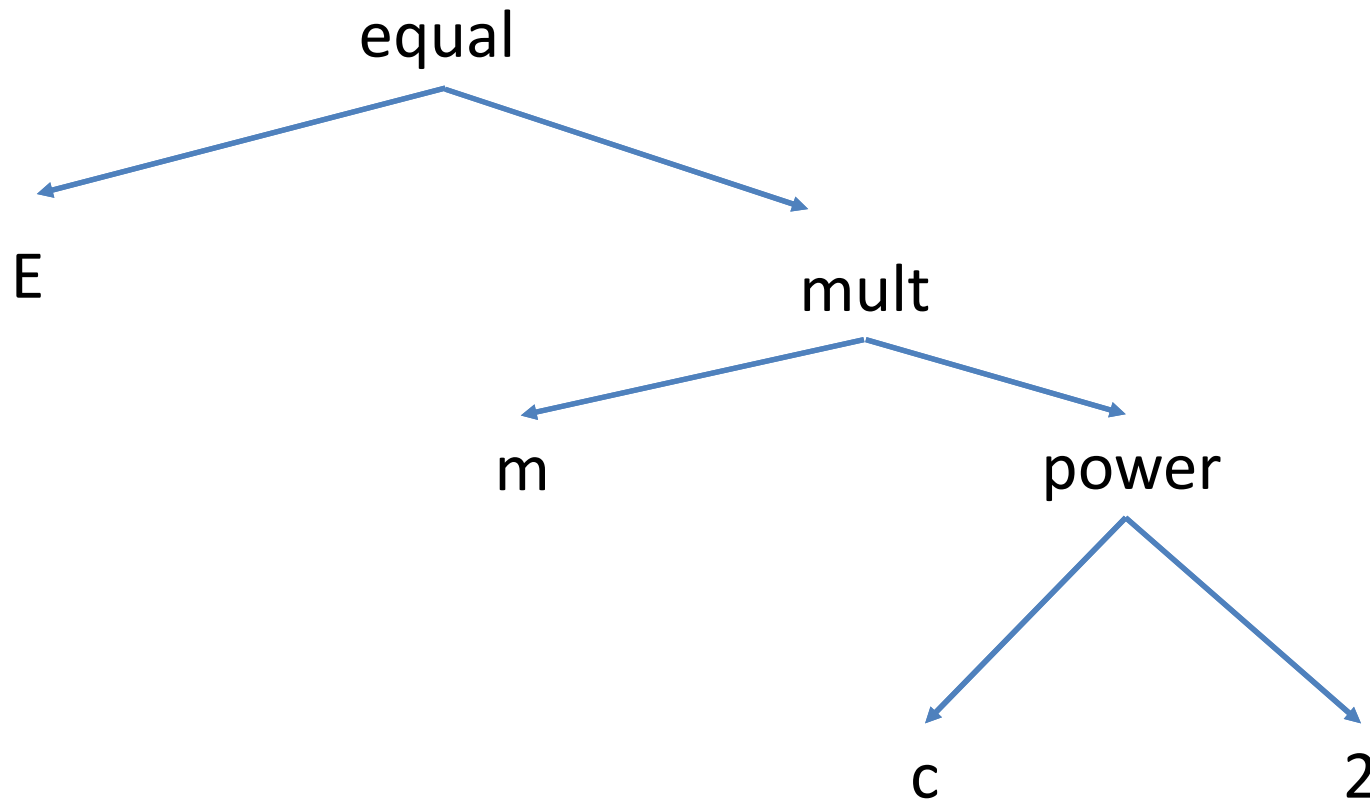
# Abstract Syntax Tree

$E = mc^2$

`equal(E, mult(m, power(c, 2)))`

# Abstract Syntax Tree

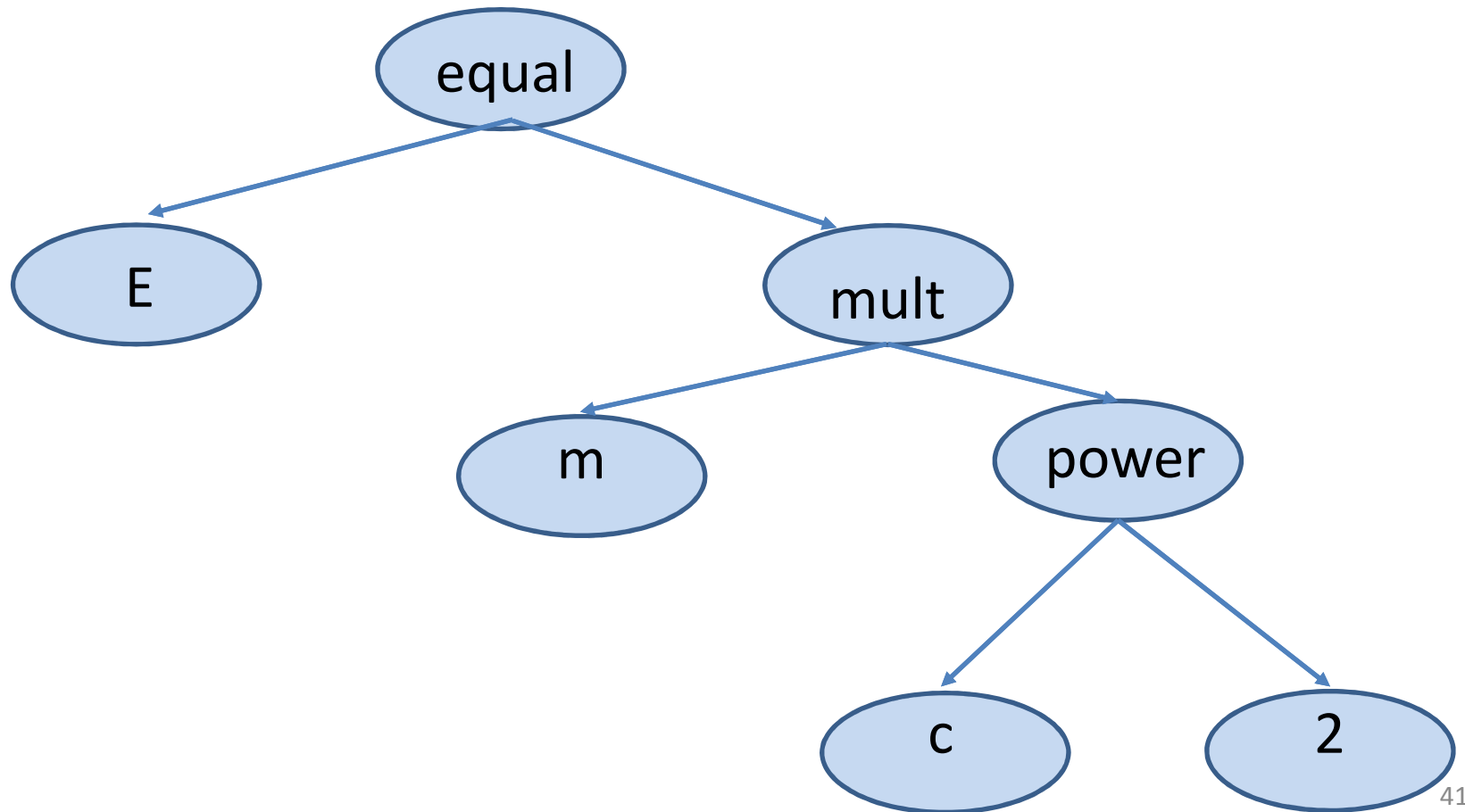
`equal(E, mult(m, power(c, 2)))`





# Abstract Syntax Tree

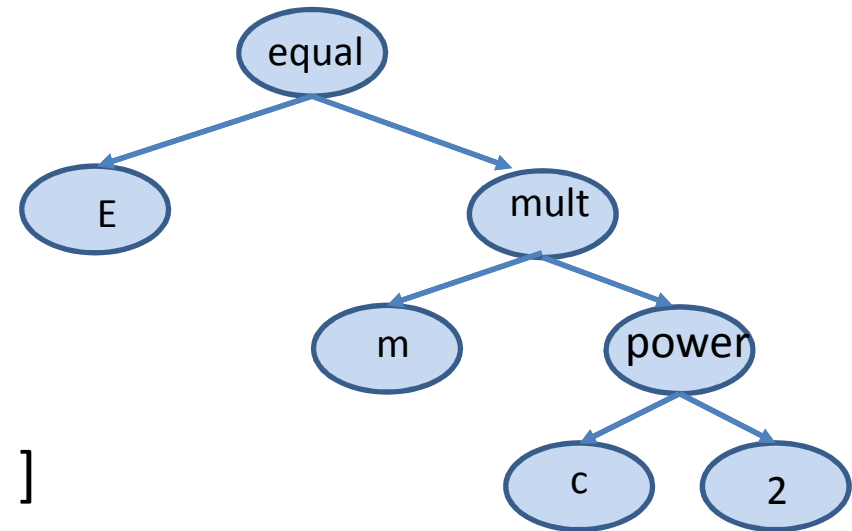
`equal(E, mult(m, power(c, 2)))`



# Abstract Syntax Tree

equal(E, mult(m, power(c, 2)))

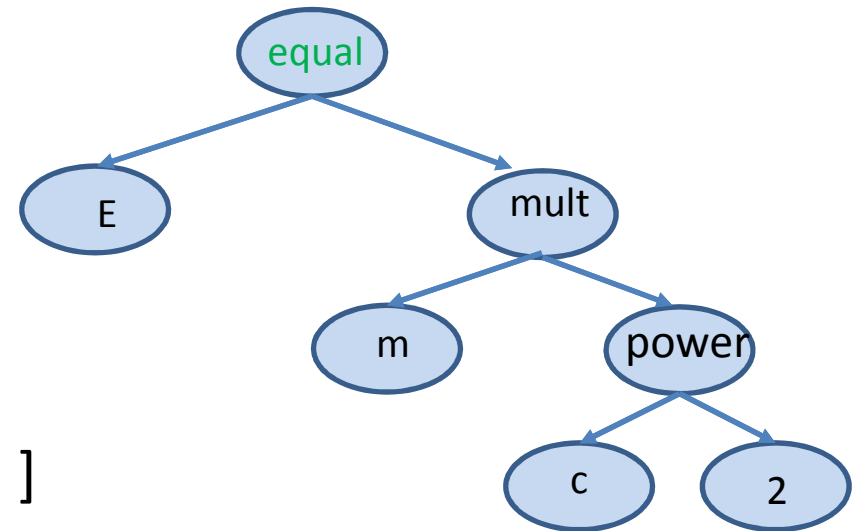
```
[ a st:Equal ; st:args (  
  [ a st:Variable ; st:name "E" ]  
  [ a st:Mult ; st:args (  
    [ a st:Variable ; st:name "m" ]  
    [ a st:Power ; st:args (  
      [ a st:Variable ; st:name "c" ]  
      [ a st:Constant ; st:value 2 ]  
    )]  
  )]  
)]
```



# Abstract Syntax Tree

equal(E, mult(m, power(c, 2)))

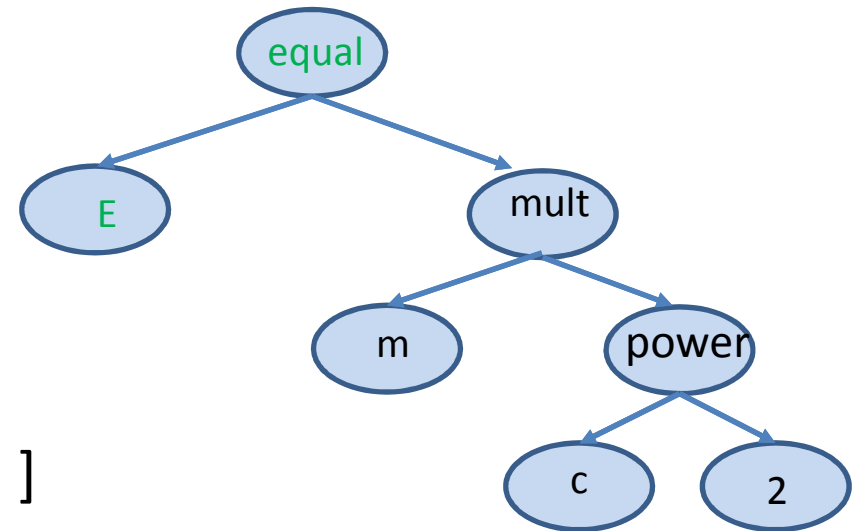
```
[ a st:Equal ; st:args (  
  [ a st:Variable ; st:name "E" ]  
  [ a st:Mult ; st:args (  
    [ a st:Variable ; st:name "m" ]  
    [ a st:Power ; st:args (  
      [ a st:Variable ; st:name "c" ]  
      [ a st:Constant ; st:value 2 ]  
    )]  
  )]  
)]
```



# Abstract Syntax Tree

equal(E, mult(m, power(c, 2)))

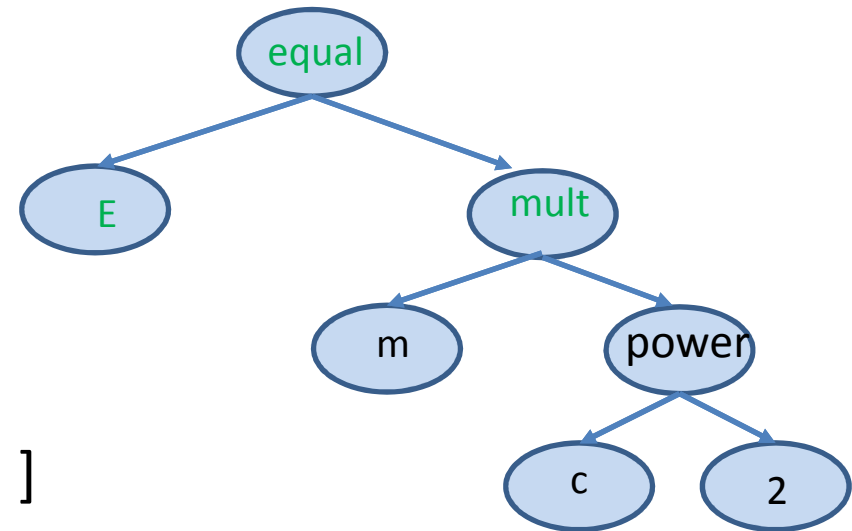
```
[ a st:Equal ; st:args (  
  [ a st:Variable ; st:name "E" ]  
  [ a st:Mult ; st:args (  
    [ a st:Variable ; st:name "m" ]  
    [ a st:Power ; st:args (  
      [ a st:Variable ; st:name "c" ]  
      [ a st:Constant ; st:value 2 ]  
    )]  
  )]  
)]
```



# Abstract Syntax Tree

equal(E, mult(m, power(c, 2)))

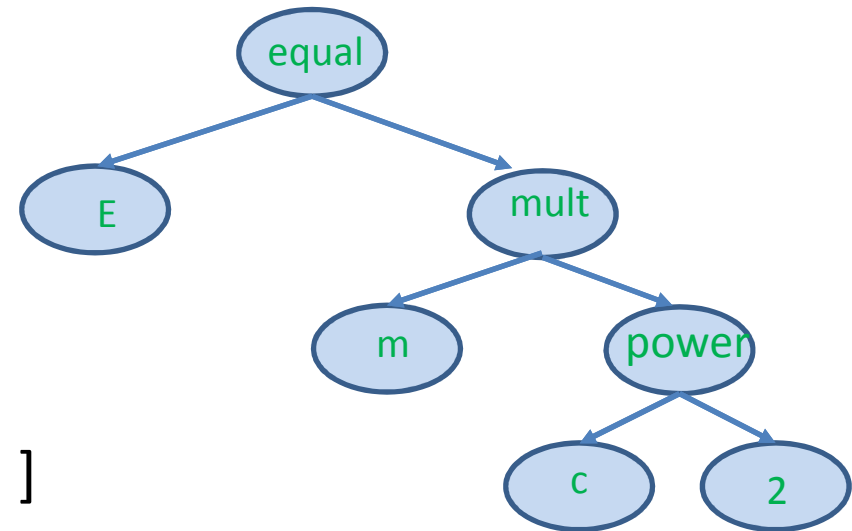
```
[ a st:Equal ; st:args (  
  [ a st:Variable ; st:name "E" ]  
  [ a st:Mult ; st:args (  
    [ a st:Variable ; st:name "m" ]  
    [ a st:Power ; st:args (  
      [ a st:Variable ; st:name "c" ]  
      [ a st:Constant ; st:value 2 ]  
    )]  
  )]  
)]
```



# Abstract Syntax Tree

equal(E, mult(m, power(c, 2)))

```
[ a st:Equal ; st:args (  
  [ a st:Variable ; st:name "E" ]  
  [ a st:Mult ; st:args (  
    [ a st:Variable ; st:name "m" ]  
    [ a st:Power ; st:args (  
      [ a st:Variable ; st:name "c" ]  
      [ a st:Constant ; st:value 2 ]  
    )]  
  )]  
)]
```



# Plan

1. RDF : Resource Description Framework
- 2. RDFS : RDF Schema**
3. SPARQL : RDF Query Language

# RDFS

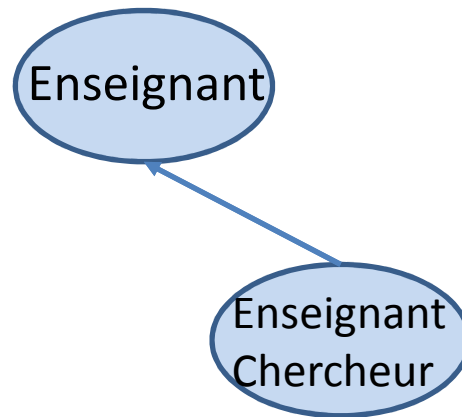
1. RDF Schema
2. Héritage multiple & multi instanciation
3. Topic, classe ou instance
4. Bonnes pratiques



# RDFS

- Définir le vocabulaire des triples RDF
- Hiérarchie de classes
- Hiérarchie de propriétés
- Signature de propriétés
- RDFS entailment (inférences RDFS)

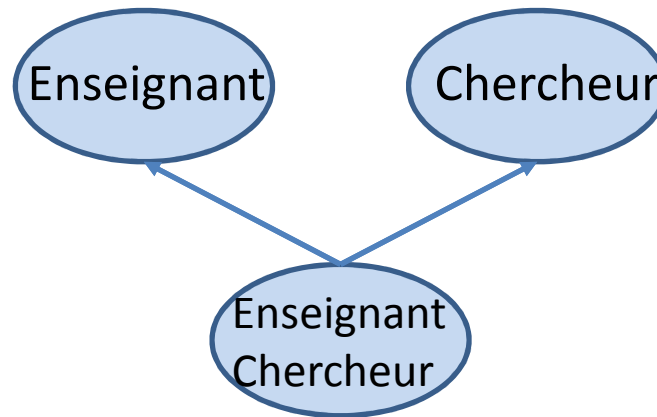
# Hiérarchie de classes



```
us:EnseignantChercheur a rdfs:Class;  
rdfs:subClassOf us:Enseignant .
```

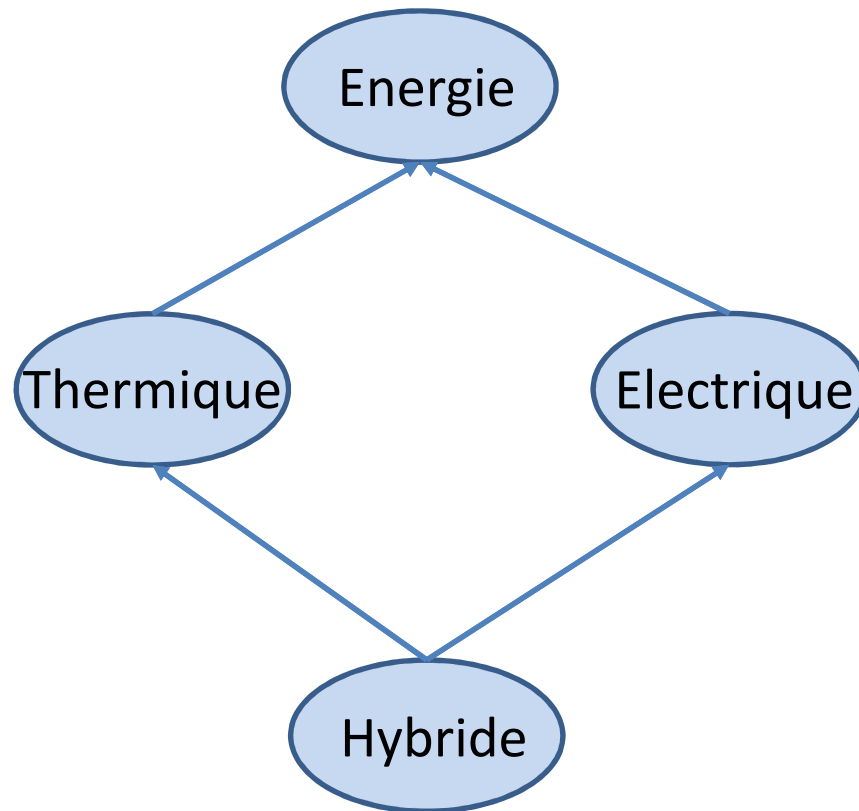
# « Héritage multiple »

- Une classe peut spécialiser plusieurs classes



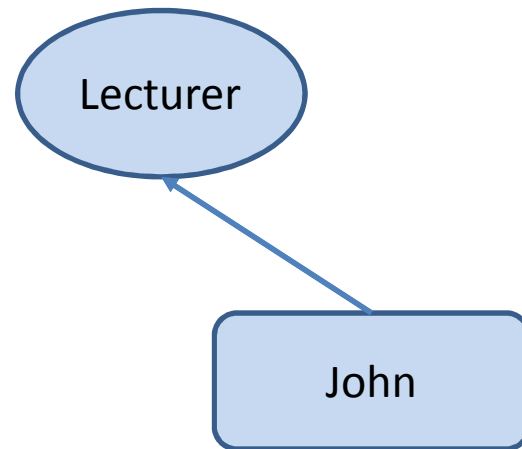
us:EnseignantChercheur **rdfs:subClasssOf**  
us:Enseignant, us:Chercheur .

# « Héritage multiple »



# Instanciación

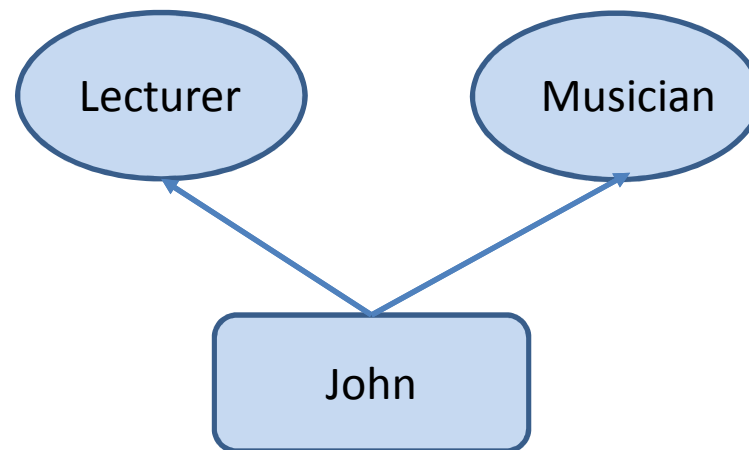
- Une ressource peut avoir un type



ex:John **rdf:type** ex:Lecturer .

# Multi instantiation

- Une ressource peut avoir plusieurs types



ex:John **rdf:type** ex:Lecturer, ex:Musician .

# Classe ou Instance

ex:Month a rdfs:Class

(1) i:January a ex:Month

(1) i:February a ex:Month

(2) ex:January rdfs:subClassOf ex:Month

(2) ex:February rdfs:subClassOf ex:Month

(2.1) [ a ex:January ]

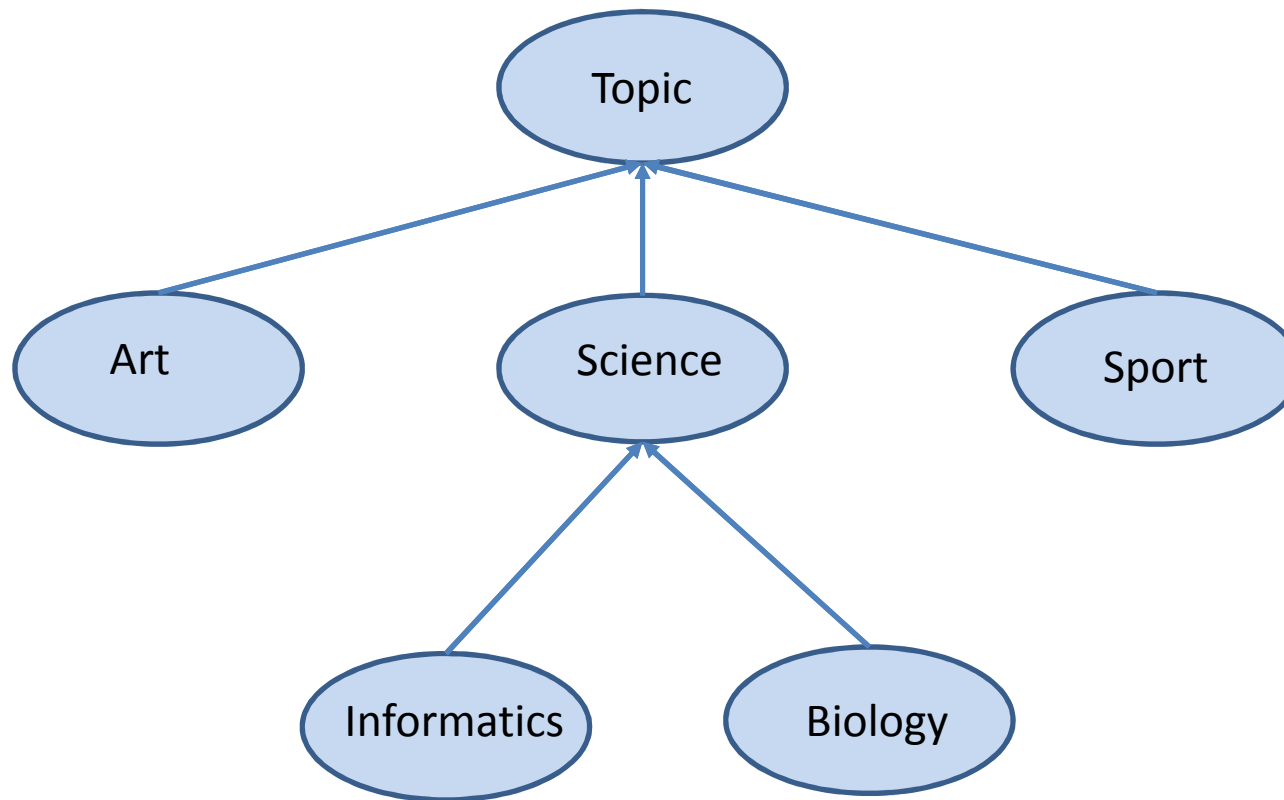
(2.2) i:January a ex:January

# Topic

- Hiérarchie de sujets pour annoter des ressources
- Exemple : annoter des documents avec leur sujet

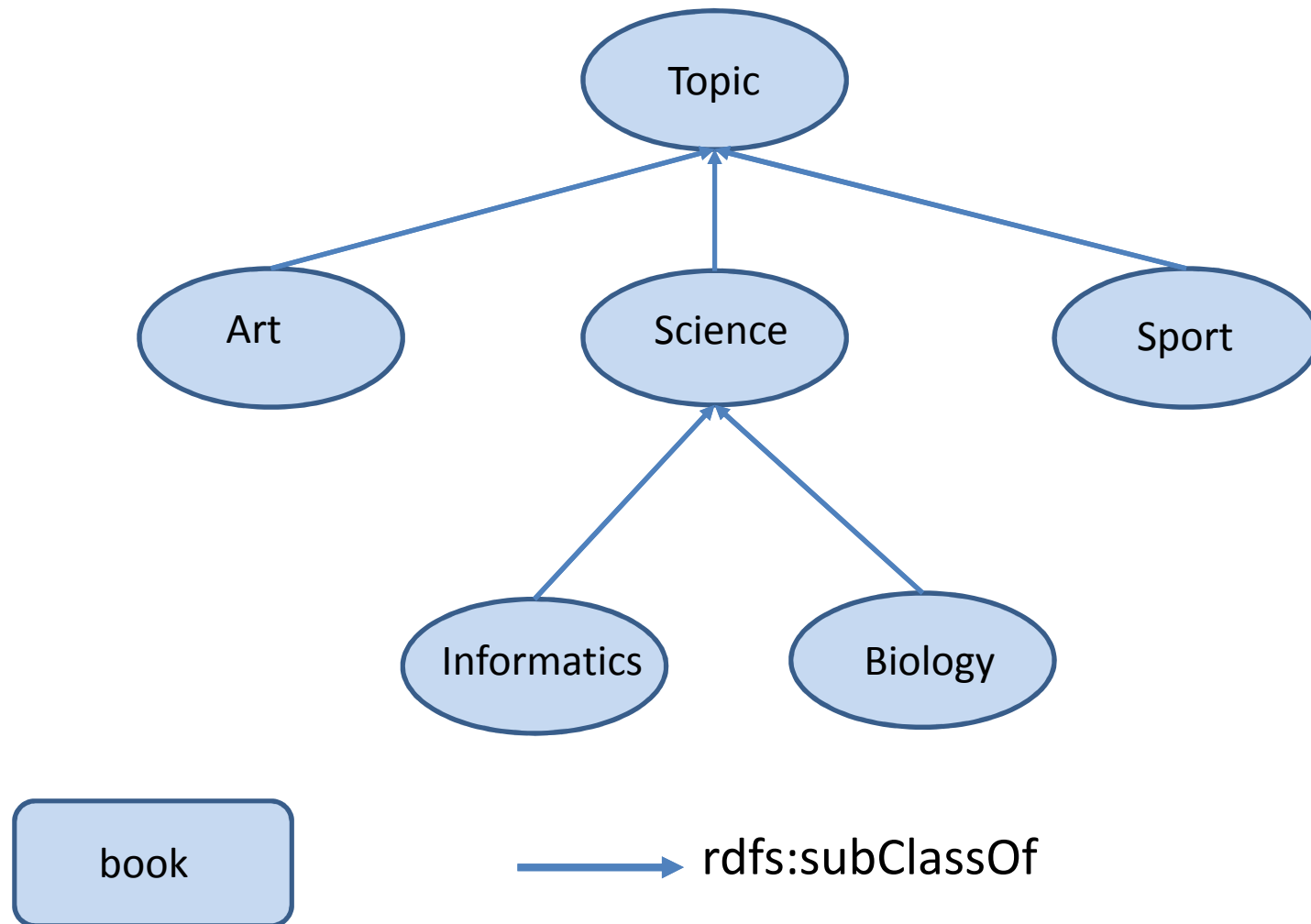


# Topic

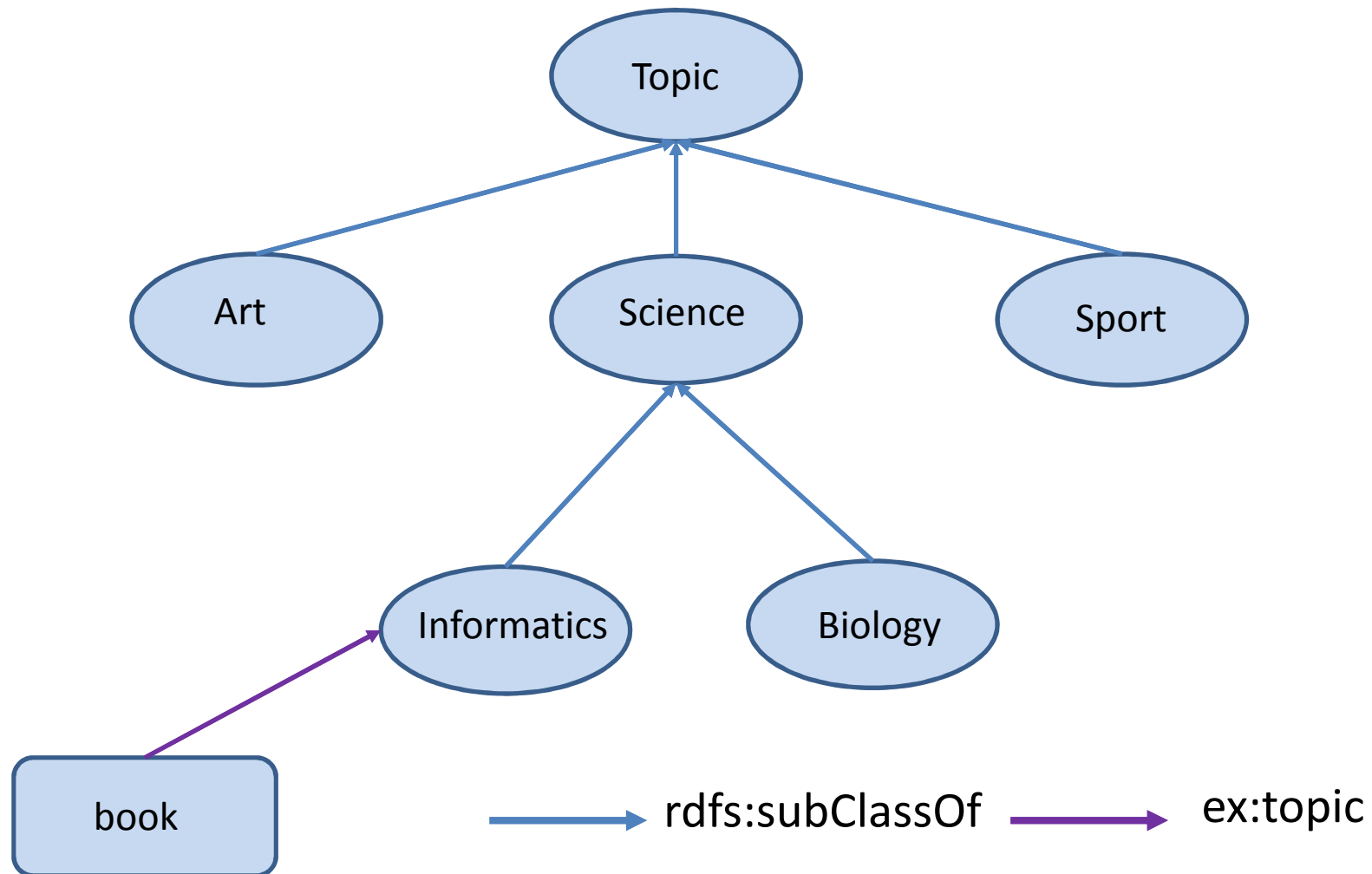


→ `rdfs:subClassOf`

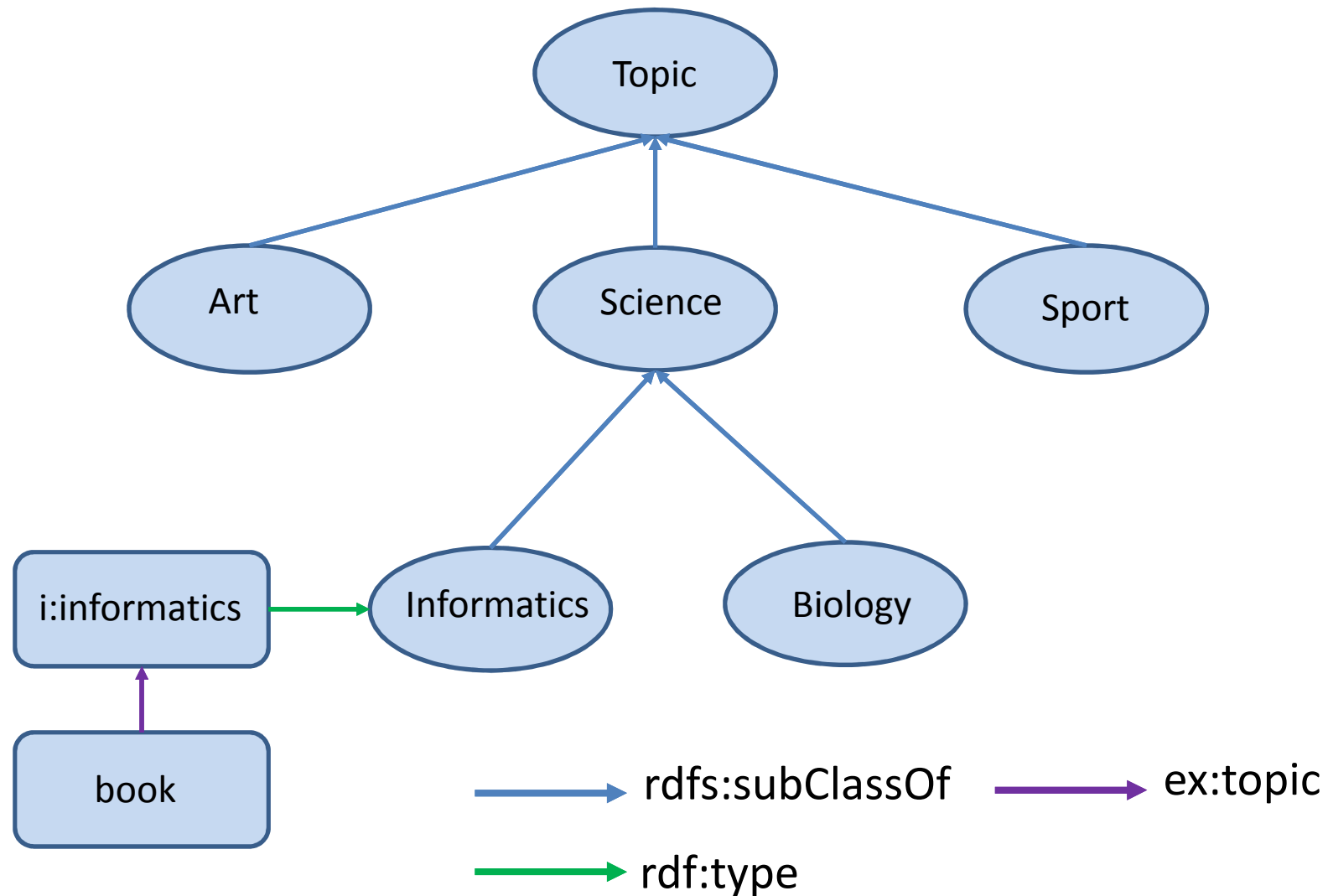
# Topic



# Topic (1)



# Topic (2)



# Héritage

- Il n'y a pas d'héritage avec RDF/S

# Bonnes pratiques RDFS

1. Définir un namespace URI et un préfixe
2. Namespace URI référence le schema
3. Définir une racine pour la hiérarchie de classes
4. Renseigner `rdfs:label` et `rdfs:comment`
5. Eviter de mélanger les classes et les propriétés

# Bonnes pratiques RDFS

1. Choisir une langue pour les noms des classes et des propriétés (e.g. tout en anglais)
2. Nom de classe au singulier avec une majuscule
  - ex:Human, ex:WebOfData
3. Nom de propriété avec une minuscule
  - ex:name, ex:firstName

# Modélisation RDF(S)

Trois « Axes » de modélisation

1. Subsumption (classe-classe)

- `rdfs:subClassOf`

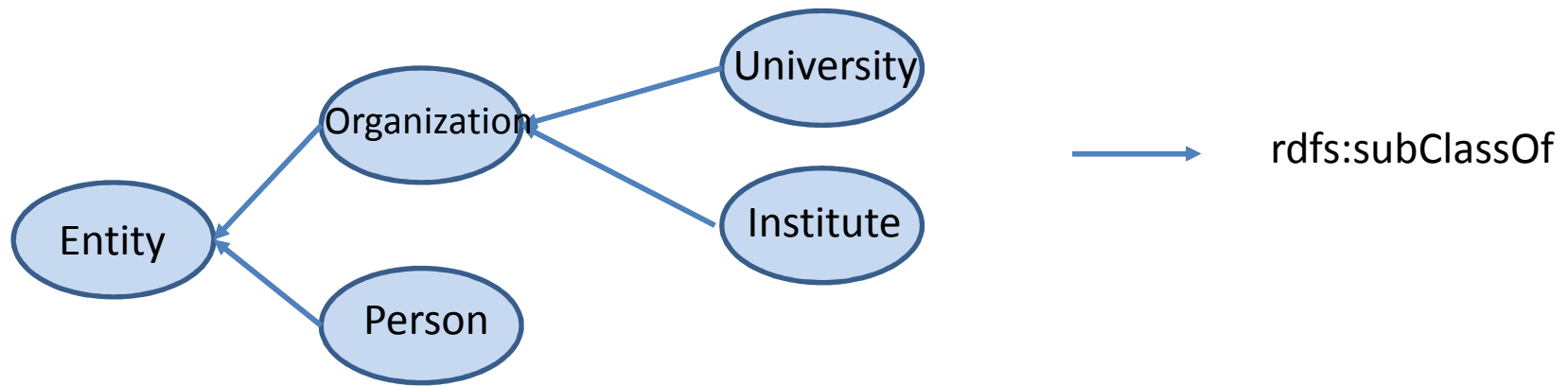
2. Instanciation (instance-classe)

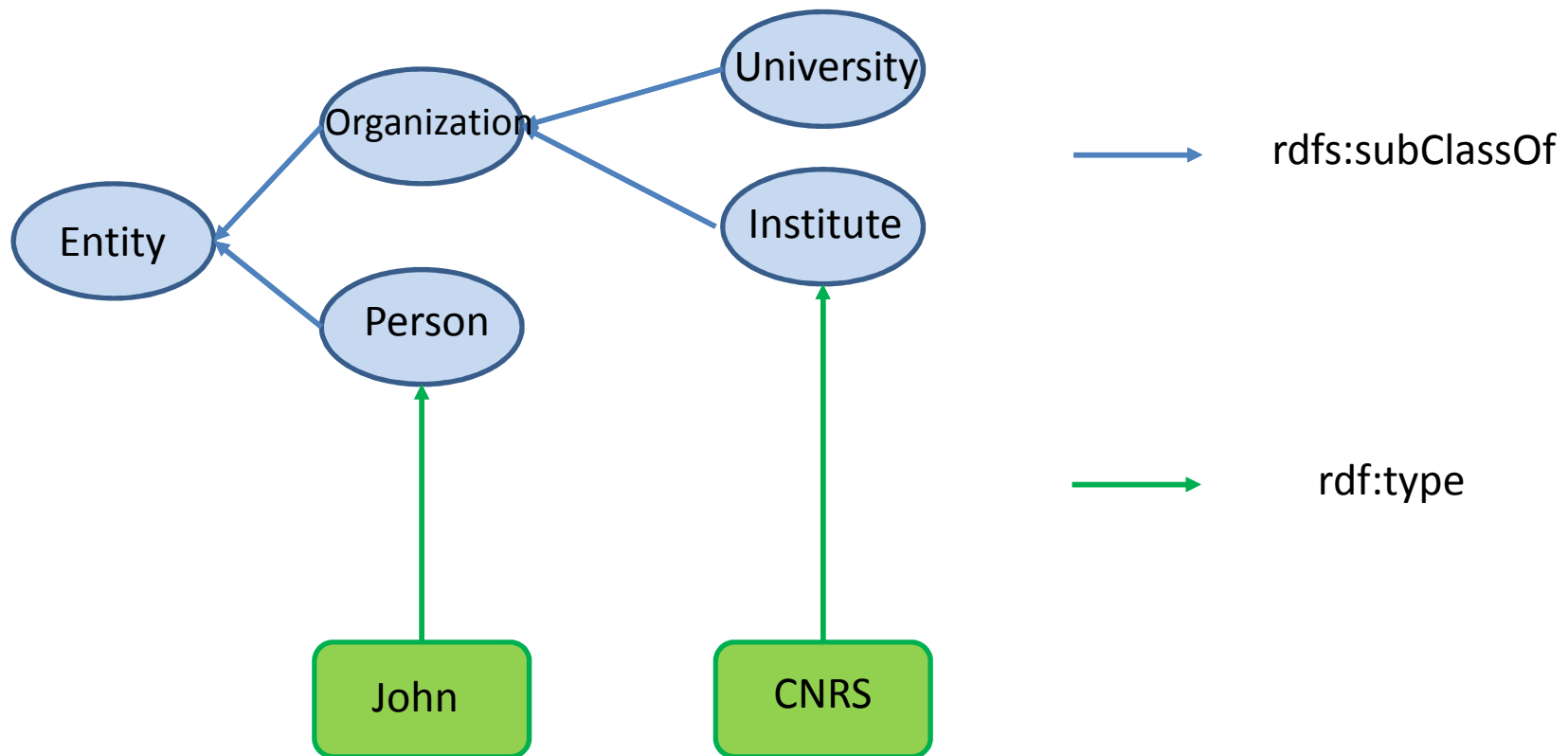
- `rdf:type`

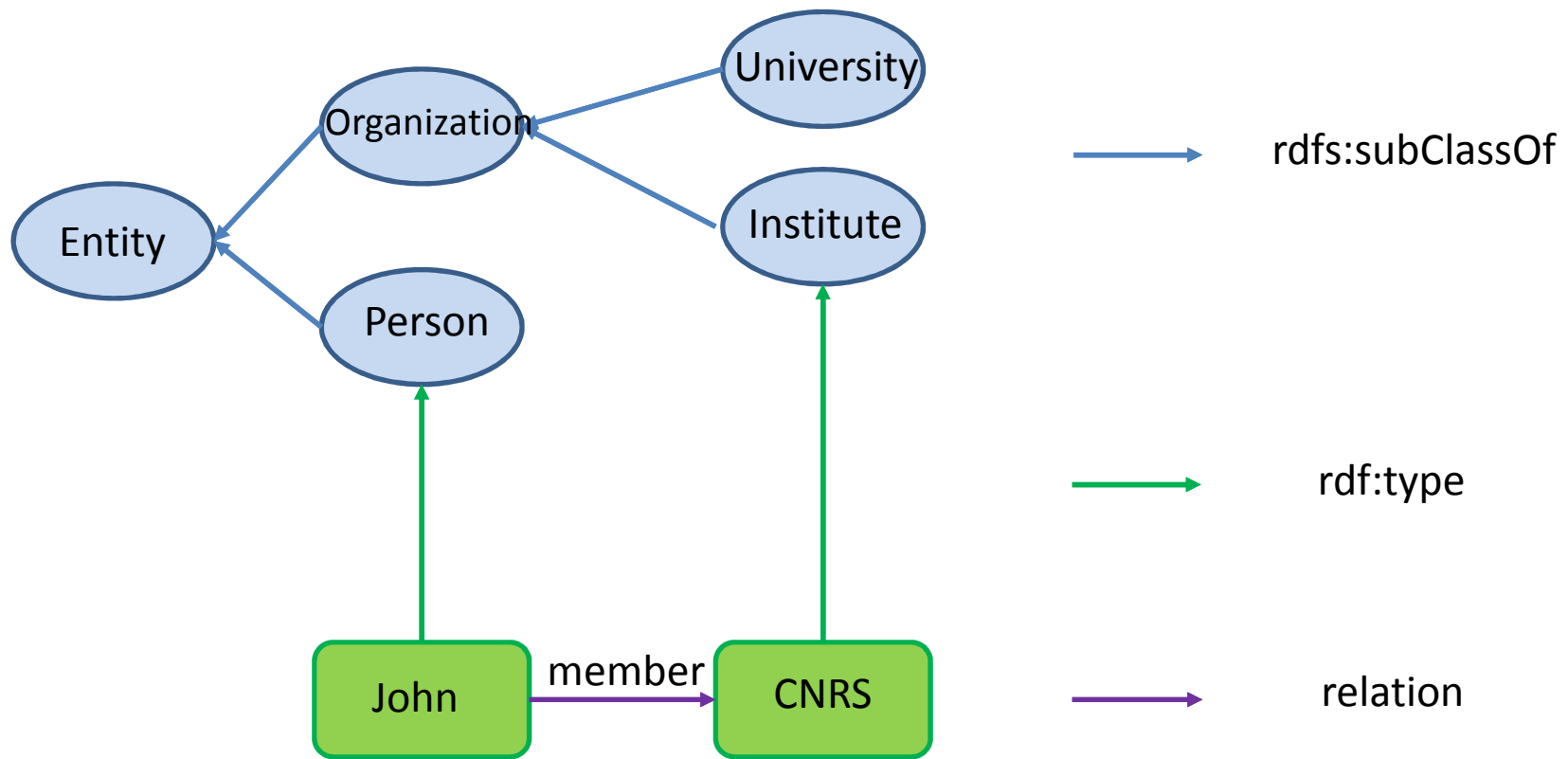
3. Relation (instance-instance)

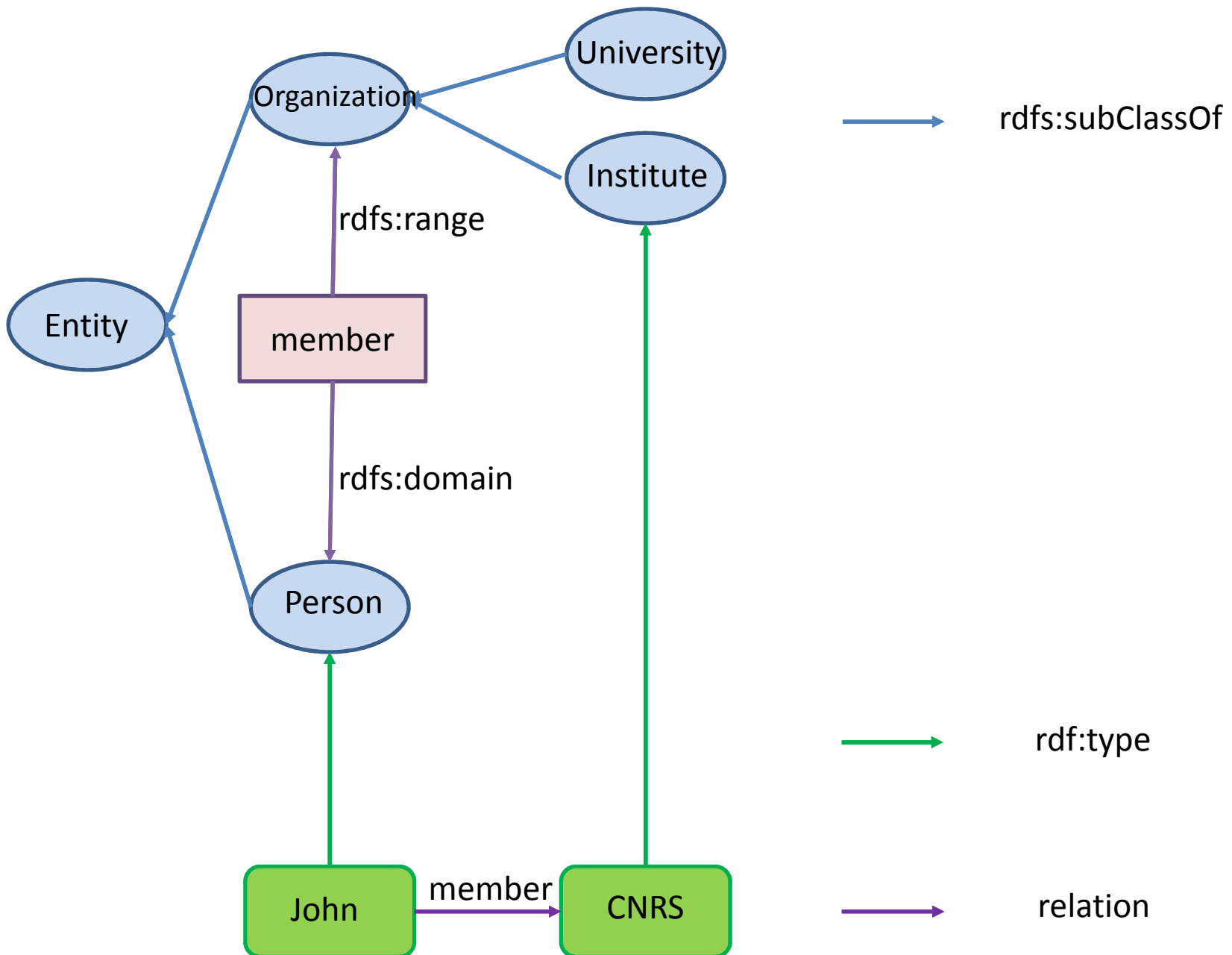
- Relation (entre ressource)
- Attribut (valeur littérale)

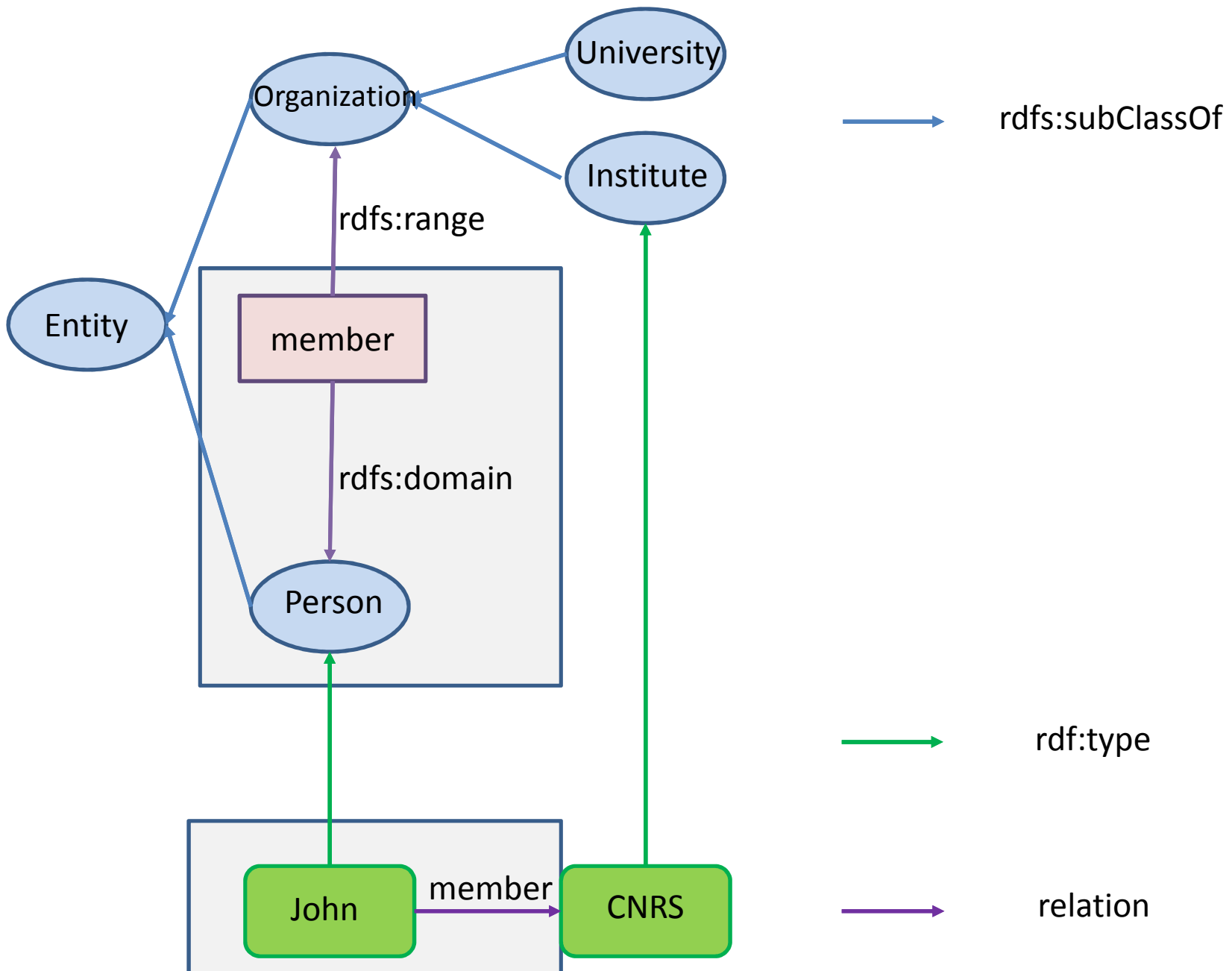


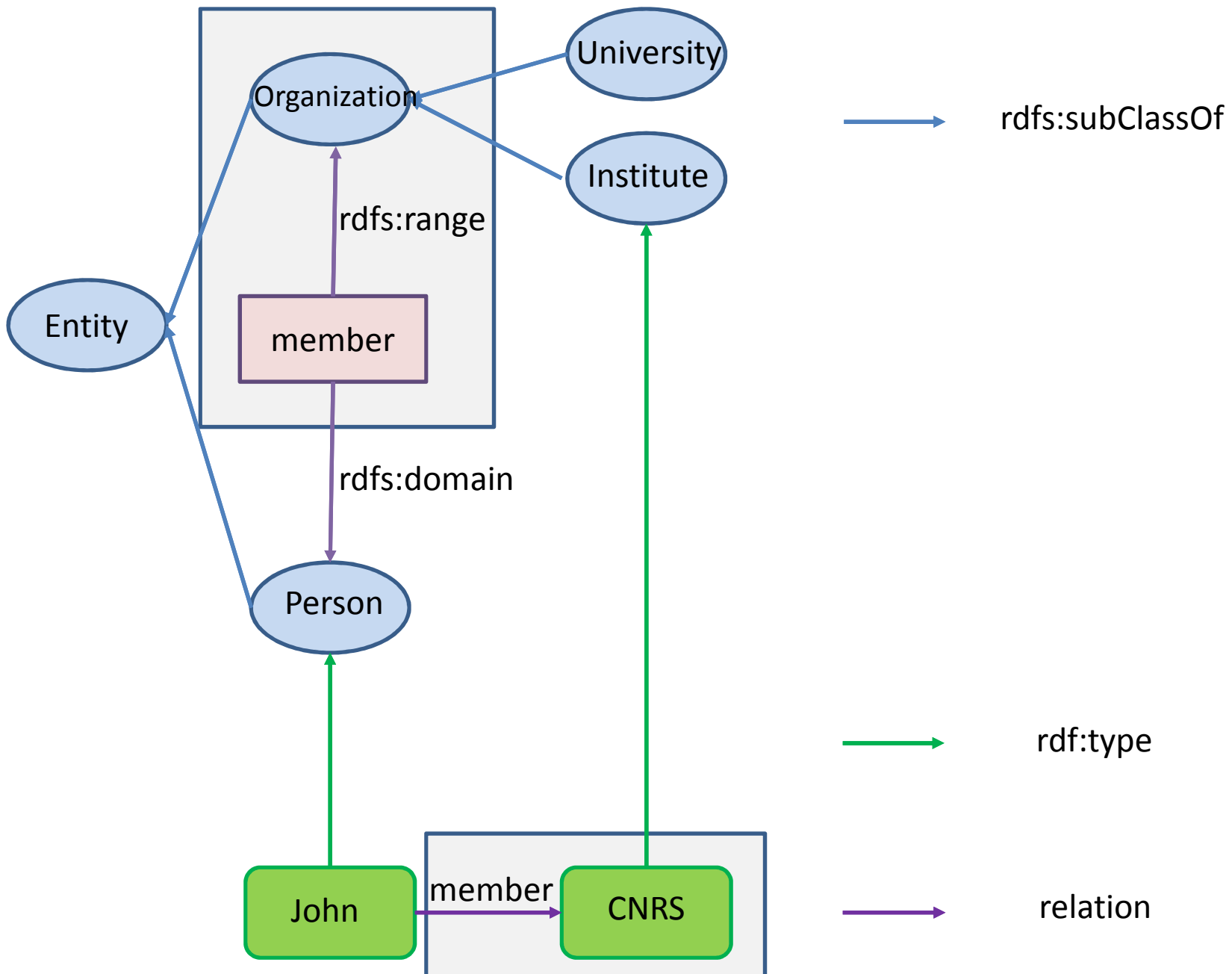


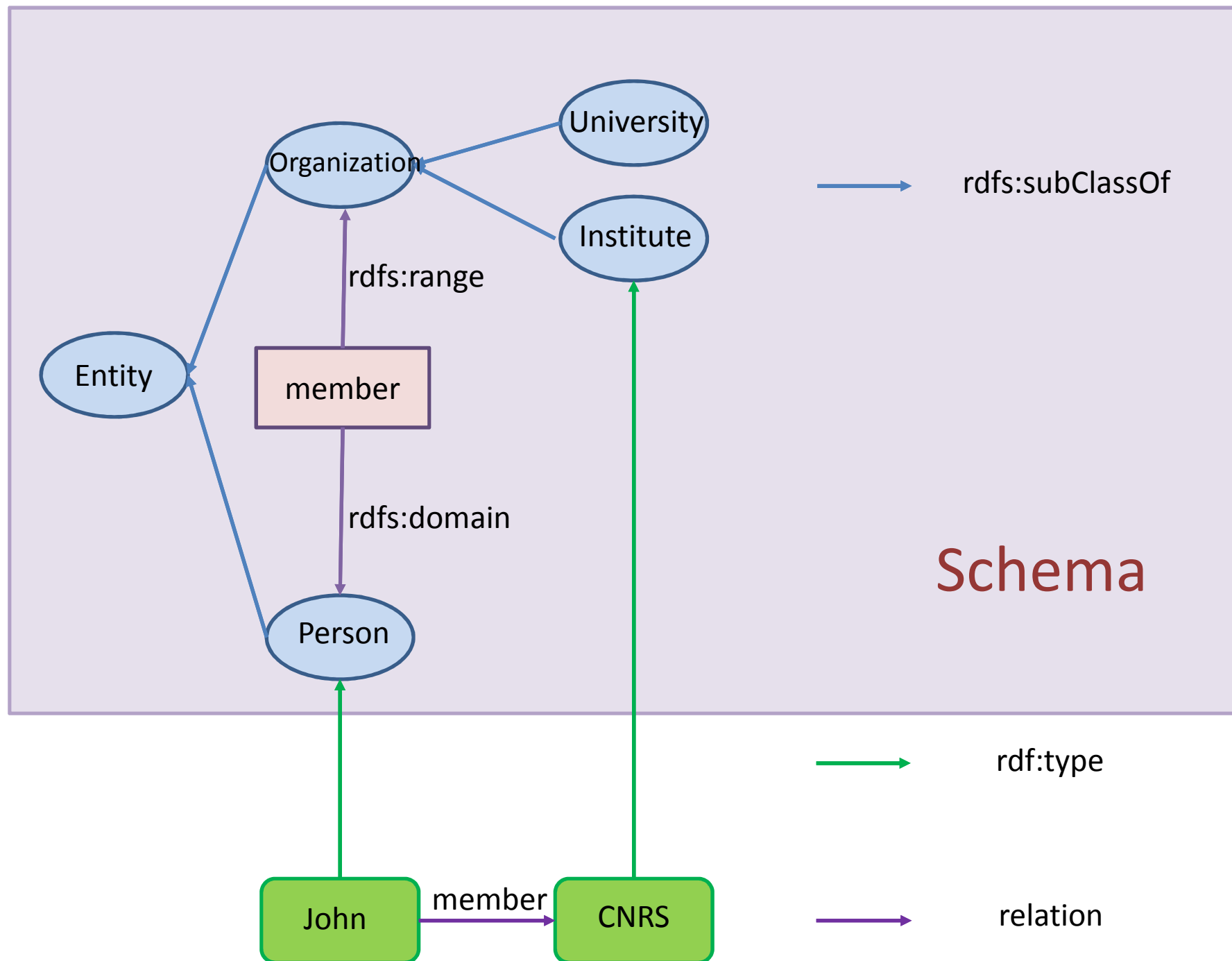


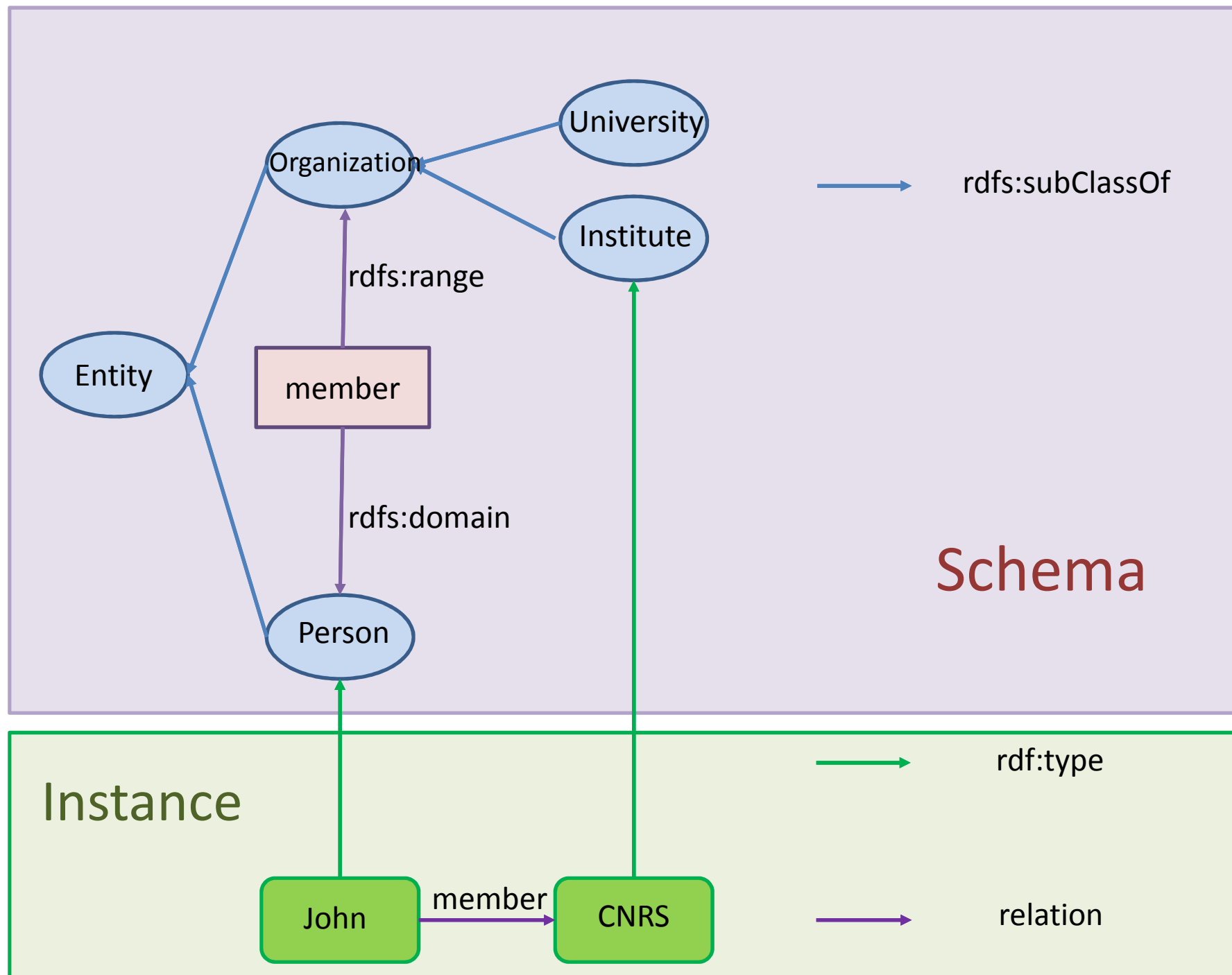






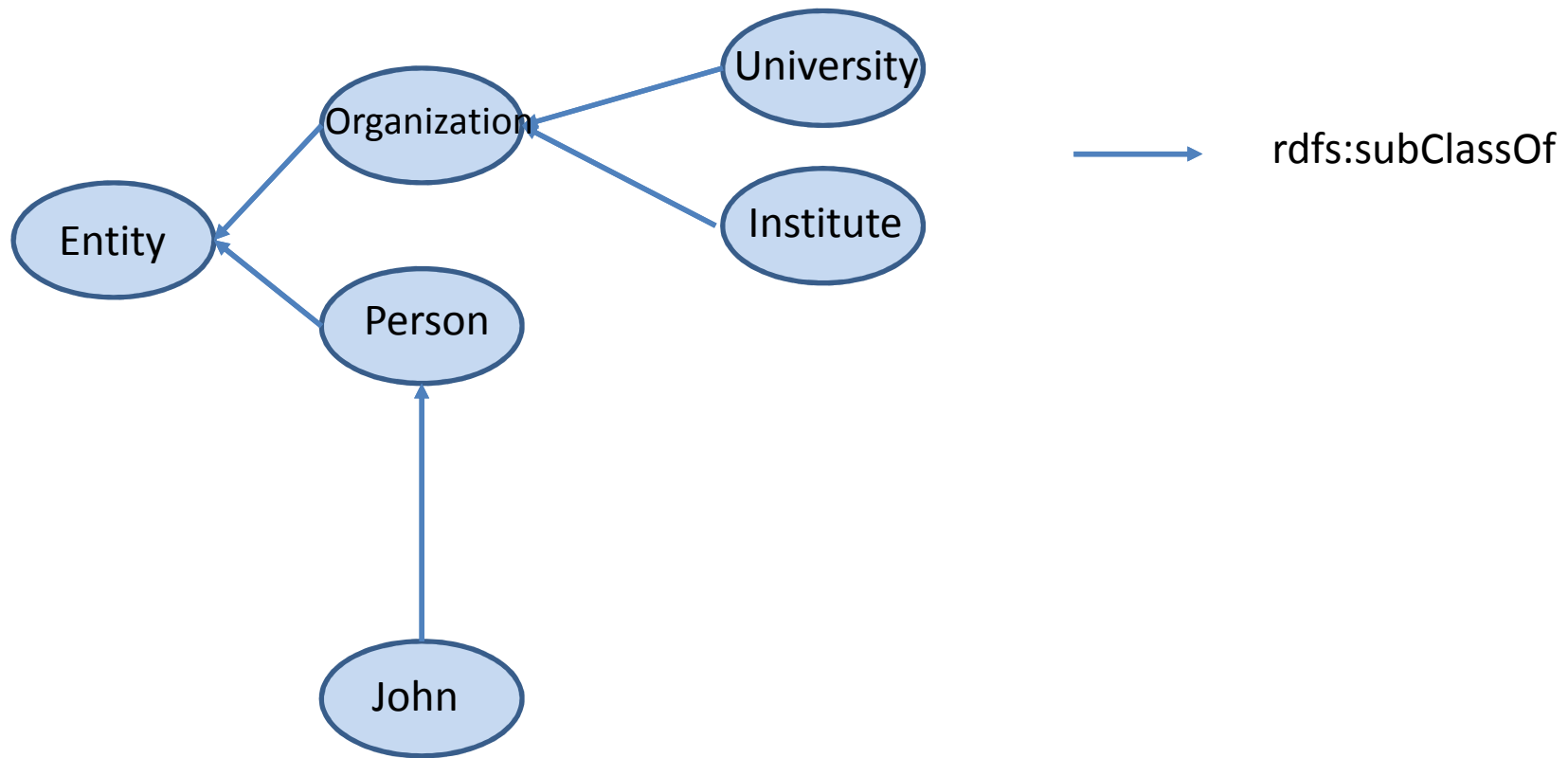


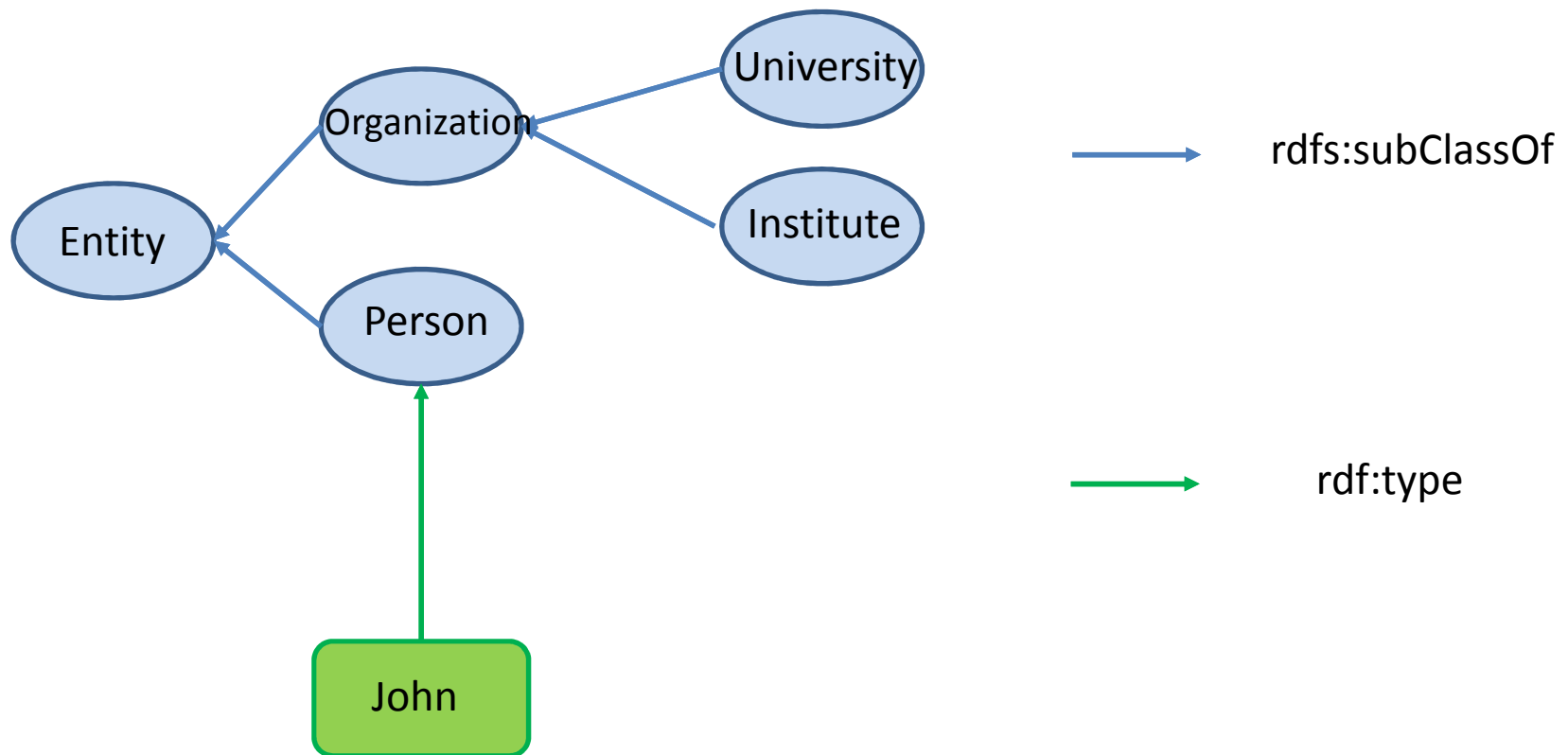




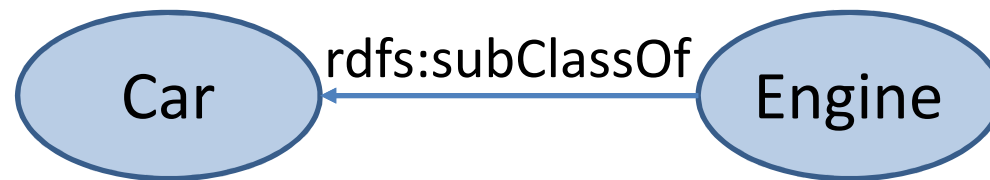


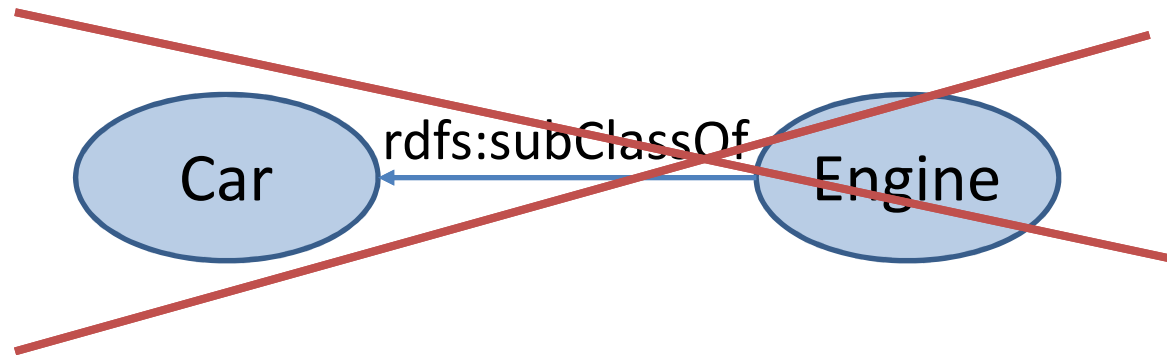
# Erreur à ne pas commettre





# Autre erreur





# RDFS, what else ?

- OWL : Web Ontology Language
  - Description Logics
- SKOS : Simple Knowledge Organization System
  - Thesaurus
- SHACL : Shapes Constraint Language
  - Constraint Language for RDF
- RIF : Rule Interchange Format
  - Inference Rules

# Plan

1. RDF : Resource Description Framework
2. RDFS : RDF Schema
3. **SPARQL : RDF Query Language**

# SPARQL

- RDF Query Language
- RDF Update Language
- Protocole HTTP

# Emuler la subsumption

```
select * where {  
    ?x rdf:type/rdfs:subClassOf* ex:Human  
}
```



# Parcourir une liste

```
select * where {  
    ?list rdf:rest*/rdf:first ?elem  
}
```

# Graphe nommé

```
select * where {
```

```
    graph ?g { ?doc ex:subject ?topic }
```

```
}
```

# Graphe nommé

```
select * where {  
    ?g ex:date "1932-06-06"^^xsd:date  
    graph ?g { ?doc ex:subject ?topic }  
}
```

# Service

Interroger SPARQL endpoint distant

```
prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
prefix o:    <http://dbpedia.org/ontology>
```

```
select * where {
  ?x o:birthPlace ?uri

  service <http://fr.dbpedia.org/sparql> {
    ?uri geo:lat ?lat ; geo:long ?lon
  }
}
```

# Conclusion

- Web de données
- Ecosystème de langages standards
- Représentation et échange de données et de connaissances sur le Web

# Conclusion

- Nombreuses bases existantes
  - <http://fr.dbpedia.org/sparql>
  - <http://data.bnf.fr/sparql>
  - <http://rdf.insee.fr/sparql>
- <https://www.w3.org/wiki/SparqlEndpoints>
- <http://lov.okfn.org/dataset/lov/>

*« Tout doit être le plus simple possible, mais pas plus simple que ça »*

Albert Einstein